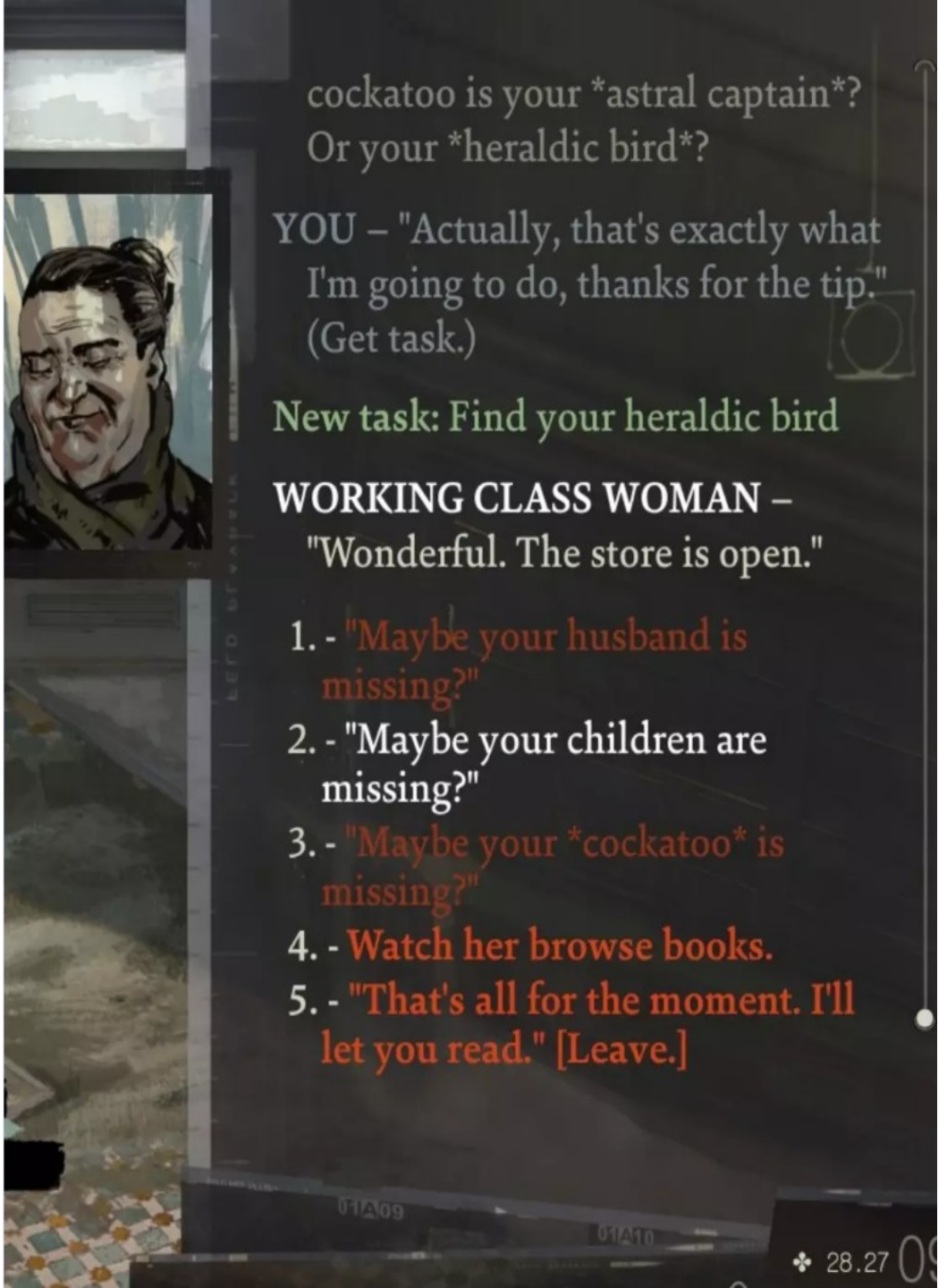
Dialogue in games - Overview

- dialogue in video games (not only 3D),
- three categories of dialogue: ambient, interactive, and cutscenes,
- Context-Aware Character Dialog in The Last of Us
- Dialogue Systems in Unity

Purposes of Dialogue

- Characterisation
- Worldbuilding
- Pacing
- Humor
- Progression



Three Categories of Dialogue

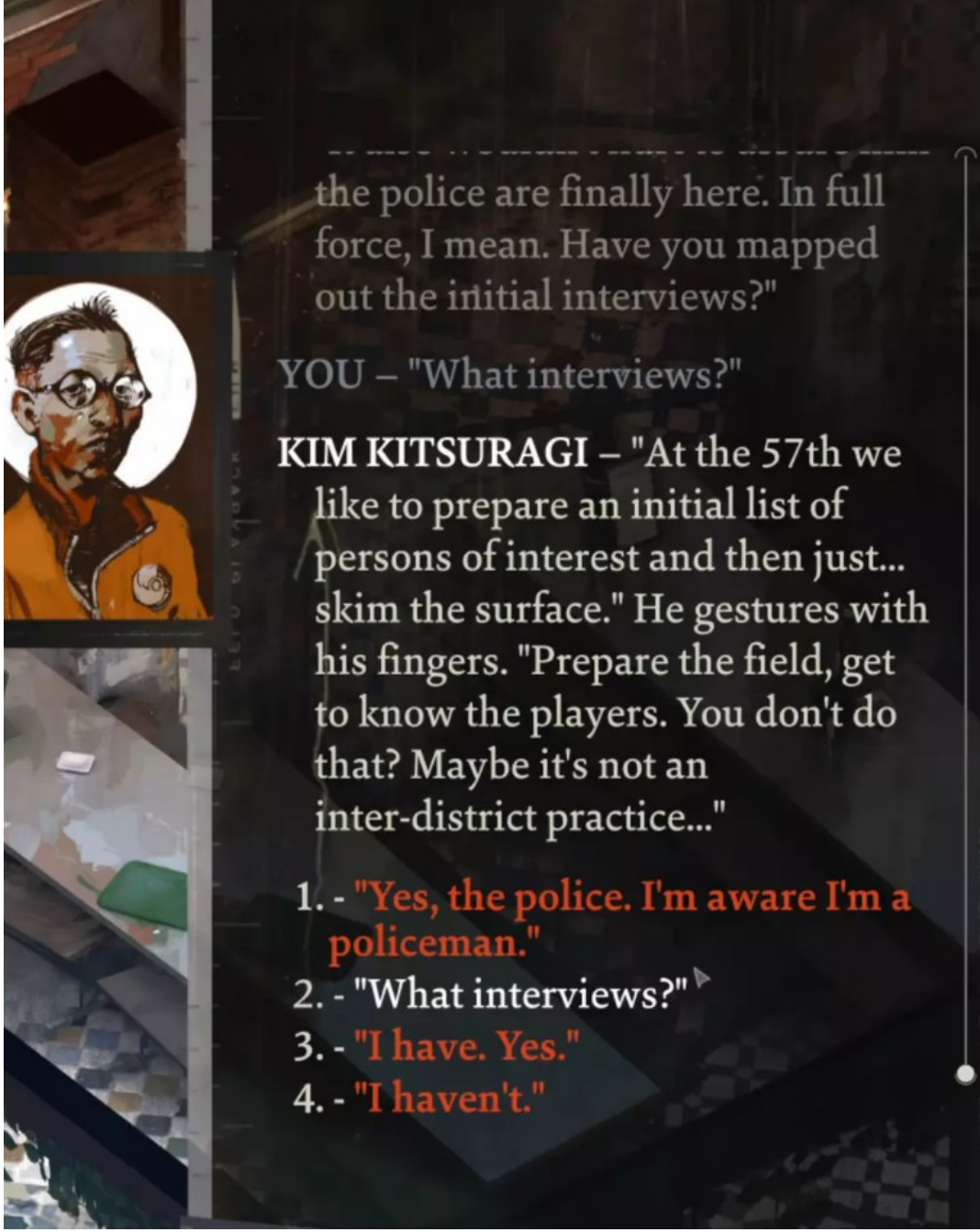
- Ambient Dialogue
- Interactive Dialogue
- Cutscenes

Types of Ambient Dialogue

- Onomatopes/ Grunts
- Barks/Soundsets
- Generic Ambients
- Customized Ambients
- Generic Banter
- Cunstomized Banter

Interactive Dialogue

- Different ways to write interactive dialogue
- Scripted conversations
- Dialog trees
- Condistional branches and exchanges



Quest-Giving Dialogue

- Opening hook: attract the player to enter dialogue
- First NPC Hub: quest-giver explains the problem
- Further NPC Hubs: details of the quest are explained
- Player Hubs: player asks questions; accepts or refuses quest
- Final NPC Hub: immediate goal is spelled out, sending the player forward

Quest-Giving Dialogue

- Success Dialogue (NPC): if player completes quest
- Failure Dialogue (NPC): if player fails or refuses quest

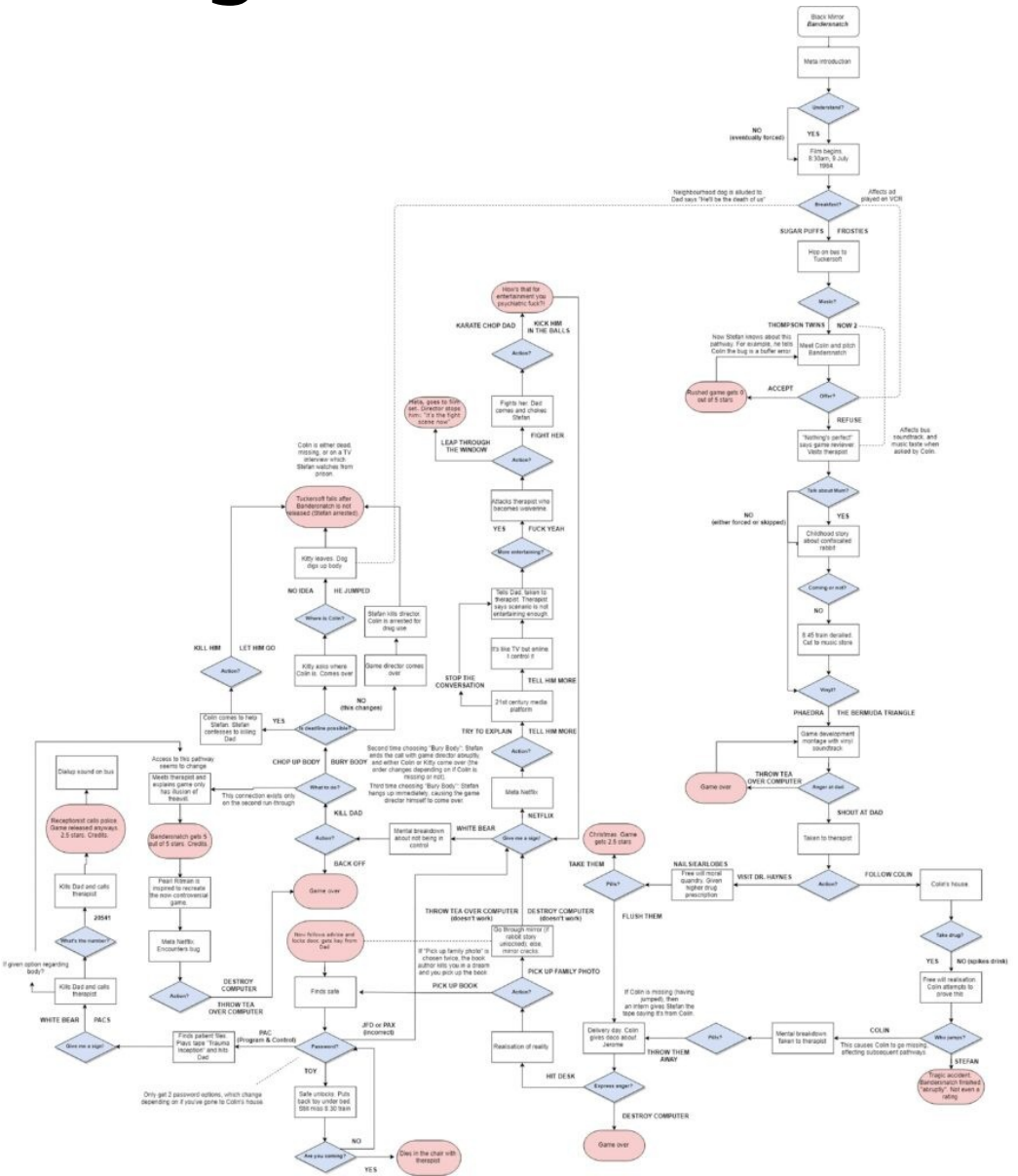
Quest-Giving Dialogue

- Common genre of game writing; often in portfolios of game writers
- Usually spoken by NPC: introduces new objective, search area, and enemies/obstacles

Quest-Giving Dialogue

- Often divided into two parts:
  - quick context and introduction to the quest
  - fleshes out the context and urges player to complete the quest
- Success or failure: what the NPC says when the player successfully completes the quest, fails it or refuses it.

Dialogue Tree



Dialogue Tree (Player ↔️ NPC)

- NPC (Guard):
  - Halt. The city gates are closed to outsiders.

Player Choices (Node 1)

1 “Why are the gates closed?”

2 “I have business inside.”

3 “Stand aside, or else.”

Branch 1: Asking for Information

- Player:
  - Why are the gates closed?
- Guard:
  - Bandits attacked last night. Orders from the captain—no entry without permission.
- ➡️ Returns to Player Choices (Node 2)
  - “Who can give permission?”
  - “Sounds dangerous. Goodbye.”

Game Check: Charisma ≥ 5 ?

- Success
  - Hm… all right. Be quick about it.
  - Outcome: Gate opens → City unlocked
- Failure
  - Rules are rules. Come back with proper papers.
  - Outcome: Quest hint added: Find city permit

Same Tree, Simplified as Structure (Pseudo-Logic)

```
Start
 └─ Guard: "The gates are closed."
     ├─ Ask why
     │   └─ Learn about bandits → more questions
     ├─ Claim business
     │   └─ Charisma check
     │       ├─ Success → Gate opens
     │       └─ Failure → New quest
     └─ Threaten
         └─ Combat or arrest
```

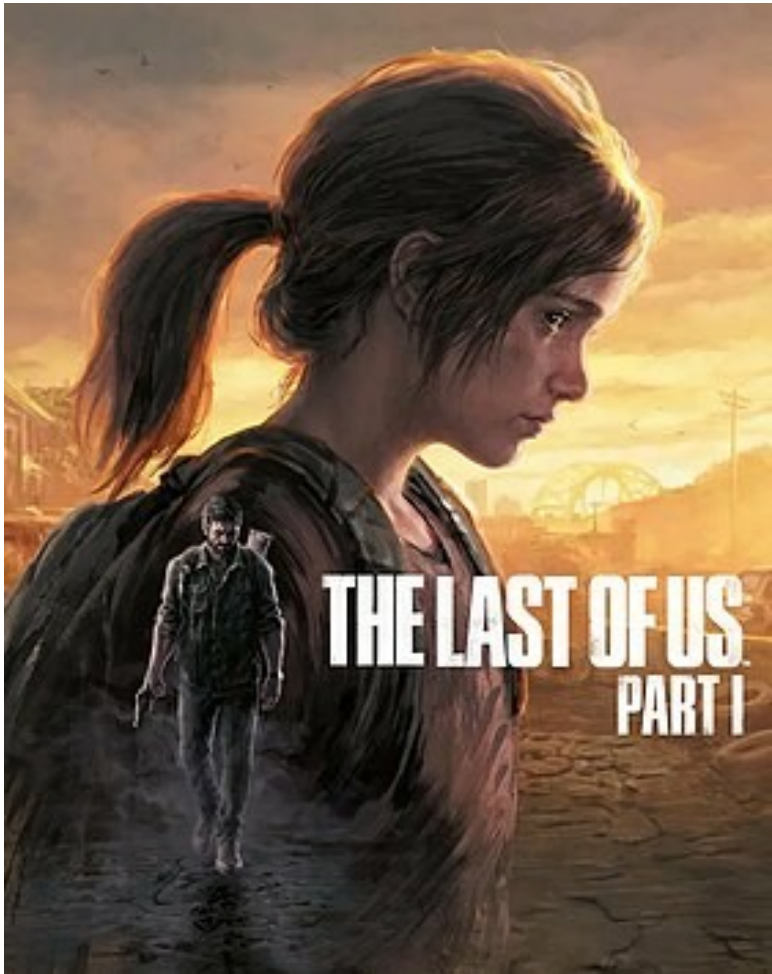
Why This Is a “Good” Dialogue Tree

- **Player agency** – different tones, different outcomes
- **Reactivity** – skills and choices matter
- **Narrative efficiency** – one NPC delivers lore, quests, and consequences
- **Reusable structure** – easy to expand with flags, reputation, or morality

Context-Aware Character Dialog in The Last of Us

The Last of Us – Introduction

- The Last of Us is a story-driven action-adventure game set in a post-apocalyptic world.
- A deadly fungal infection has collapsed civilization, forcing survivors to fight for resources and safety.



World & Atmosphere

- Cities lie abandoned and overgrown by nature.
- The game emphasizes loneliness, danger, and environmental storytelling.

Main Characters

- Joel is a hardened survivor shaped by loss.
- Ellie is a brave teenage girl who may hold the key to humanity’s survival.
- Their emotional bond drives the story.

Infected & Threats

- The infected are humans overtaken by the Cordyceps fungus.
- Clickers are among the most dangerous enemies, hunting by sound.

Looselly Based On

- Rule Databases for Contextual Dialog and Game Logic … or … How to Make Writers Even More Awesome

Requirements for the Dialog System of The Last of Us

- Significant changes in AI for TLOU
- how the AI has changed between Uncharted and TLOU,
  - and how that affects the requirements of TLOU’s dialog system…

Dialog in *Uncharted*

Dialog in Uncharted

- AI in the Uncharted series was basically two-state:

1 Unaware of player

2 Full-out **combat**

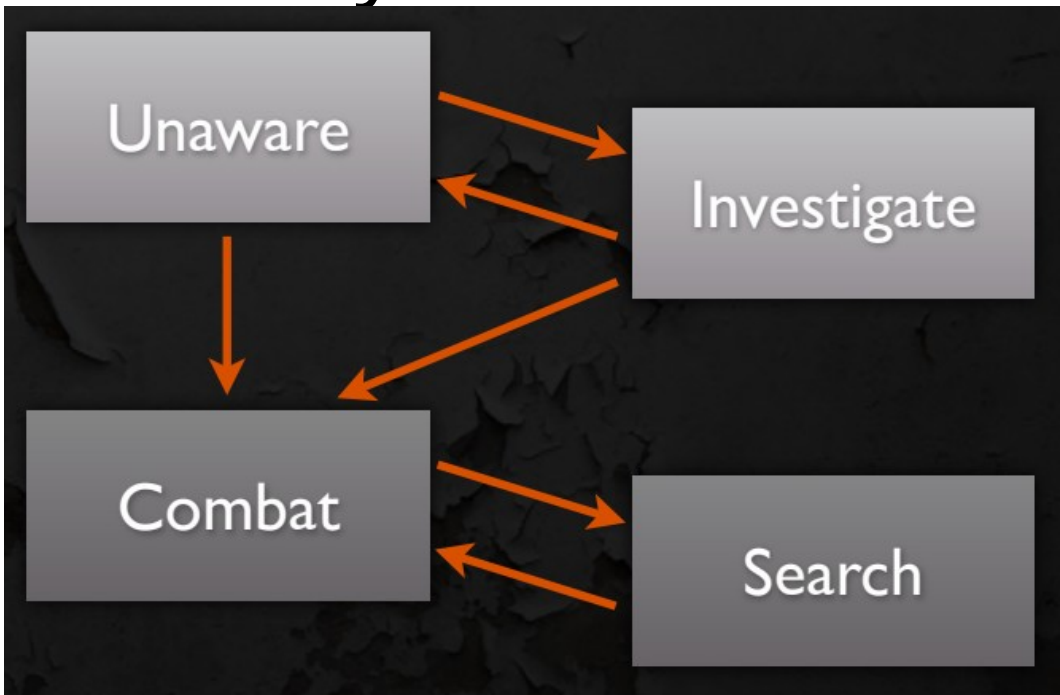
Dialog in Uncharted

- Uncharted dialog system requirements:
  - Individual characters saying single lines
    - Hand-scripted conversations during in-game cinematics (IGCs)
    - One-liner battle chatter during combat
- Enemies with unique voices
- Wide variety of lines
- Simple fire-and-forget API

TLOU Requirements

TLOU Requirements

- In The Last of Us, stealth became a core gameplay
- mechanic
- • Multi-state AI, with ability to transition freely
- between states:
- 1. Unaware
- 2. Investigate
- 3. Combat
- 4. Search



TLOU Requirements

- Telegraph AI state changes
- Reinforce helpfulness of buddy AI
- Scripted dialog (as in prior games)
  - Can be interrupted when the AI investigate a noise, or enter combat
- Dynamic conversations during combat
- Context- and location-aware dialog
- Branching conversations based on game events and player choices

Core Dialog System

- C++ programmers and scripters request a logical line of dialog, typically in response to a gameplay event
- Dialog system translates the request into a physical line of dialog to play
  - Select from multiple variants of the line
  - Avoid repetition
  - Prevent inappropriate lines from playing

Defining a Line of Dialog

```
(dialog-line 'line-out-of-ammo
  (character 'joel
    (lines
      joel-out-of-ammo-01   ;; "Dammit, I'm out!"
      joel-out-of-ammo-02   ;; "Crap, need more bullets."
      joel-out-of-ammo-03   ;; "Oh, now I'm REALLY mad."
    )
  )
  (character 'ellie
    (lines
      ell-out-of-ammo-01   ;; "Help, I'm out!"
      ell-out-of-ammo-02   ;; "Got any more ammo?"
    )
  )
  ...
```

Defining a Line of Dialog

```
(dialog-line 'line-out-of-ammo
  ...
  (character 'hunter-a
    (lines
       huna-out-of-ammo-01  ;; "I'm out!"
       huna-out-of-ammo-02  ;; "Need more ammo!"
    )
  )
  ...
  (character 'hunter-h
    (lines
       hunh-out-of-ammo-01  ;; "I'm out!"
       hunh-out-of-ammo-02  ;; "Need more ammo!"
    )
  )
)
```

Defining a Line of Dialog

- Usually best to split the definition into multiple files:
  - one for Joel
  - one for Ellie
  - one for Hunters (A through H)
  - one for Infected
- and/or split up per-level

Playing a Line of Dialog

Playing a Line of Dialog

- API to play a line of dialog:
  - Ideally one or two simple functions (exposed to both C++ and script)
  - Asynchronous (fire and forget)…
  - … or blocking (wait for line to complete before continuing)

Playing a Line of Dialog from Script

```
(define-state-script 'find-ladder-igc
  (state ('initial)
    (on (event 'ladder-seen)
      (wait-move-to 'ellie (get-locator 'ladder-01))
      (wait 0.5)
      (wait-say 'joel 'line-we-could-use-that-ladder)

      (say 'ellie 'line-ok-lemme-get-it-down)
      (wait-move-to 'ellie (get-locator 'ladder-02))
      ;; ...
    )
  )
)
```

Playing a Line of Dialog from Script

```
(define-state-script 'find-ladder-igc
  (state ('initial)
    (on (event 'ladder-seen)
      (wait-move-to 'ellie (get-locator 'ladder-01))
      (wait 0.5)
      (wait-say 'joel 'line-we-could-use-that-ladder)

      (say 'ellie 'line-ok-lemme-get-it-down)
      (wait-move-to 'ellie (get-locator 'ladder-02))
      ;; ...
    )
  )
)
```

Playing a Line of Dialog from Script

```
(define-state-script 'find-ladder-igc
  (state ('initial)
    (on (event 'ladder-seen)
      (wait-move-to 'ellie (get-locator 'ladder-01))
      (wait 0.5)
      (wait-say 'joel 'line-we-could-use-that-ladder)

      (say 'ellie 'line-ok-lemme-get-it-down)
      (wait-move-to 'ellie (get-locator 'ladder-02))
      ;; ...
    )
  )
)
```

Playing a Line of Dialog from C++

```
void Skill::OnEvent(const Event& evt)
{
    Npc* pNpc = GetSelf(); // grab a pointer to the NPC
    switch (evt.GetMessage())
    {
    case SID('player-seen'):
        // play a line of dialog...
        pNpc->Say(SID('line-player-seen'));

        // ... and move to closest cover
        pNpc->MoveTo(GetClosestCover());
        break;
    // ...
    }
    // ...
}
```

Variety

- Given a logical line with multiple, randomlychosen physical lines…
  - … we usually want to prevent repetition
  - … may also want to control probability of hearing each physical line

Line Probabilities

```
(dialog-line 'line-out-of-ammo
  (character 'joel
    (lines
      (joel-out-of-ammo-A :probability 0.20)
      (joel-out-of-ammo-B :probability 0.20)
      (joel-out-of-ammo-C :probability 0.35)
      (joel-out-of-ammo-D :probability 0.25)
    )
  )
  ...
```

Line Probabilities

- To select a line using probabilities:
- Add up the line probabilities to create a cumulative distribution function (CDF)
- Calculate a random number x such that $0 \leq x < 1$, using rand() or similar
- Search the lines until one is found whose CDF is greater than x

Logical Line Priorities

- Need to prevent inappropriate lines from playing
- e.g. saying “I’m out of ammo!” while playing a hit reaction
- Simple solution: priorities

Logical Line Priorities

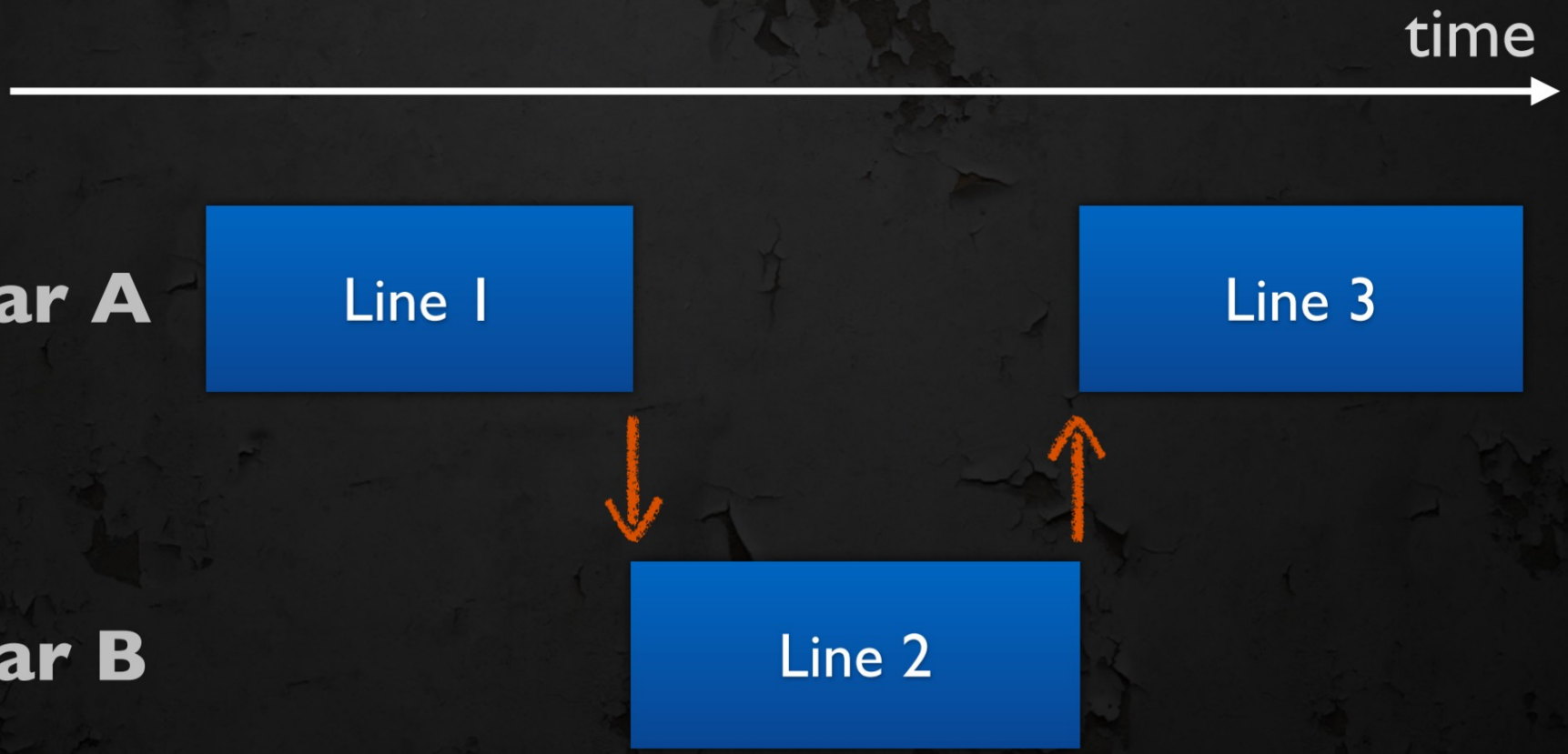
- Need to prevent inappropriate lines from playing
- e.g. saying “I’m out of ammo!” while playing a hit reaction
- Simple solution: priorities

5: death
4: hit reactions
3: IGCs
2: AI chatter
1: efforts

Logical Line Priorities

- Making an interruption sound “good” can be a bit tricky
- Fade out the interrupted dialog line?
  - NO
- Play a glottal stop sound?
  - MAYBE
- Abruptly stop the interrupted line?
  - WORKED BEST - KISS

Defining a Simple Conversation



Defining a Simple Conversation

```
(conversation-segment 'conv-searching-for-stuff-01
  (:rule []
   :line 'line-did-you-find-anything
         ;; "Hey, did you find anything?"
   :next-seg 'conv-searching-for-stuff-02
  )
)

(conversation-segment 'conv-searching-for-stuff-02
  (:rule []
   :line 'line-nope-not-yet
         ;; "I've been looking for an hour..."
   :next-seg 'conv-searching-for-stuff-03
  )
)
```

Defining a Simple Conversation

```
(conversation-segment 'conv-searching-for-stuff-01
  (:rule []
   :line 'line-did-you-find-anything
         ;; "Hey, did you find anything?"
   :next-seg 'conv-searching-for-stuff-02
  )
)

(conversation-segment 'conv-searching-for-stuff-02
  (:rule []
   :line 'line-nope-not-yet
         ;; "I've been looking for an hour..."
   :next-seg 'conv-searching-for-stuff-03
  )
```

Defining a Simple Conversation

Defining a Simple Conversation

Rules

- Basic idea:
  - Story writers and/or sound designers use simple rules to select between alternative lines of dialog at runtime…
  - … or to control branching conversations
  - The rules are written in terms of facts, which are stored in one or more dictionaries

Defining a Simple Conversation

```
[ ('health       >=  0.5)
  ('level        =   'hunter-city)
  ('enemy-count  <   3) ]
```

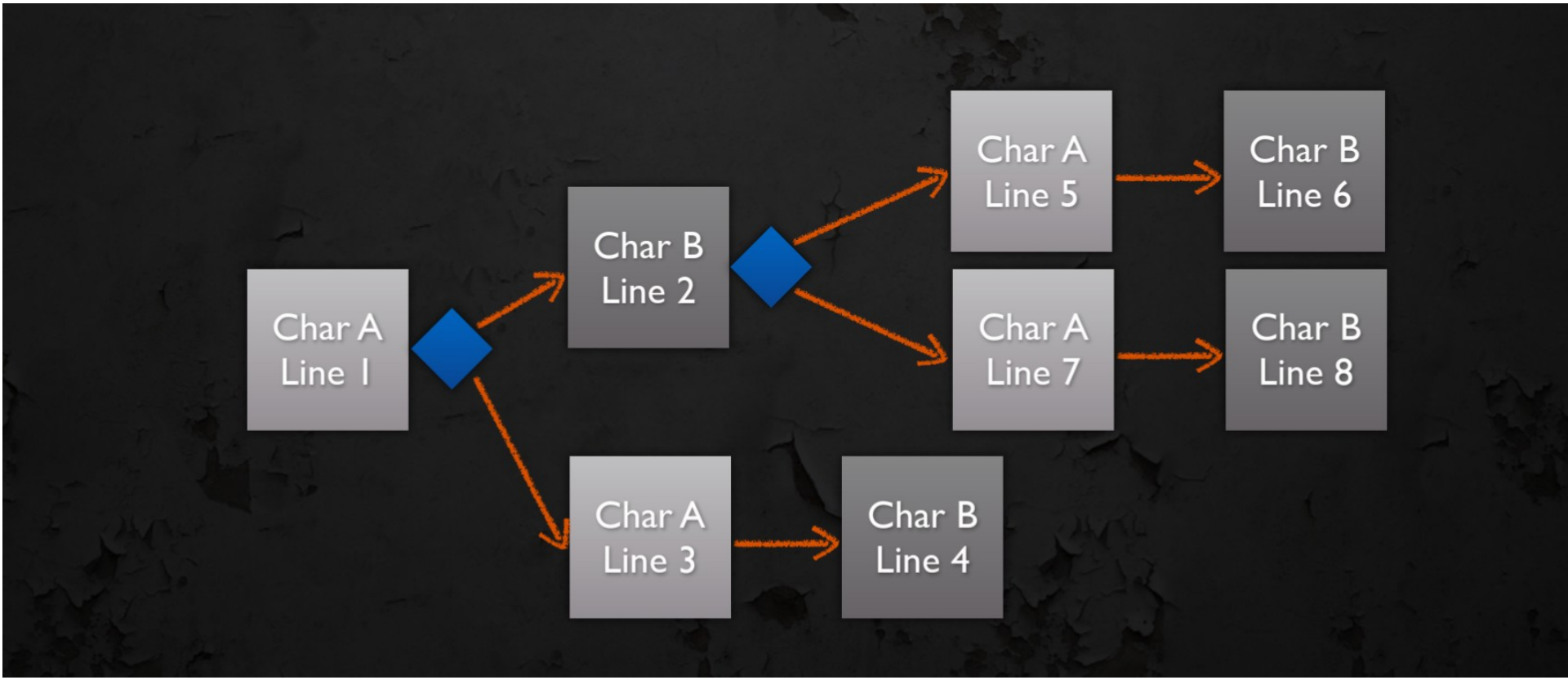
- Criteria are implicitly combined using a Boolean **AND** operator

Example

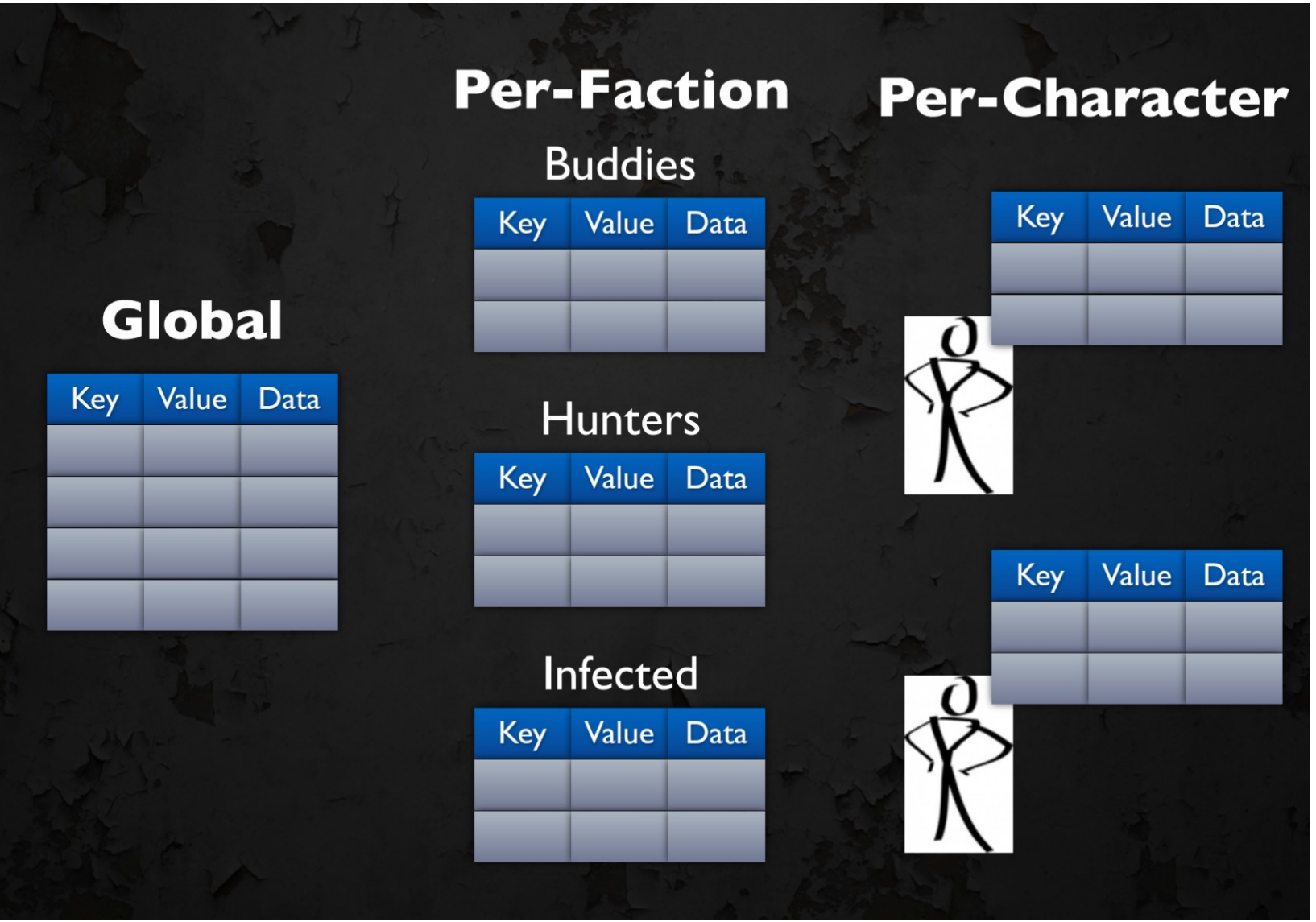
Branching Conversations

```
(conversation-segment 'conv-spot-player
  ( :rule [('speaker = 'joel)]
    :line 'line-spot-player-joel
    :next-seg 'conv-spot-player-joel
  )
  ( :rule [('speaker = 'ellie)]
    :line 'line-spot-player-ellie
    :next-seg 'conv-spot-player-ellie
  )
)
```

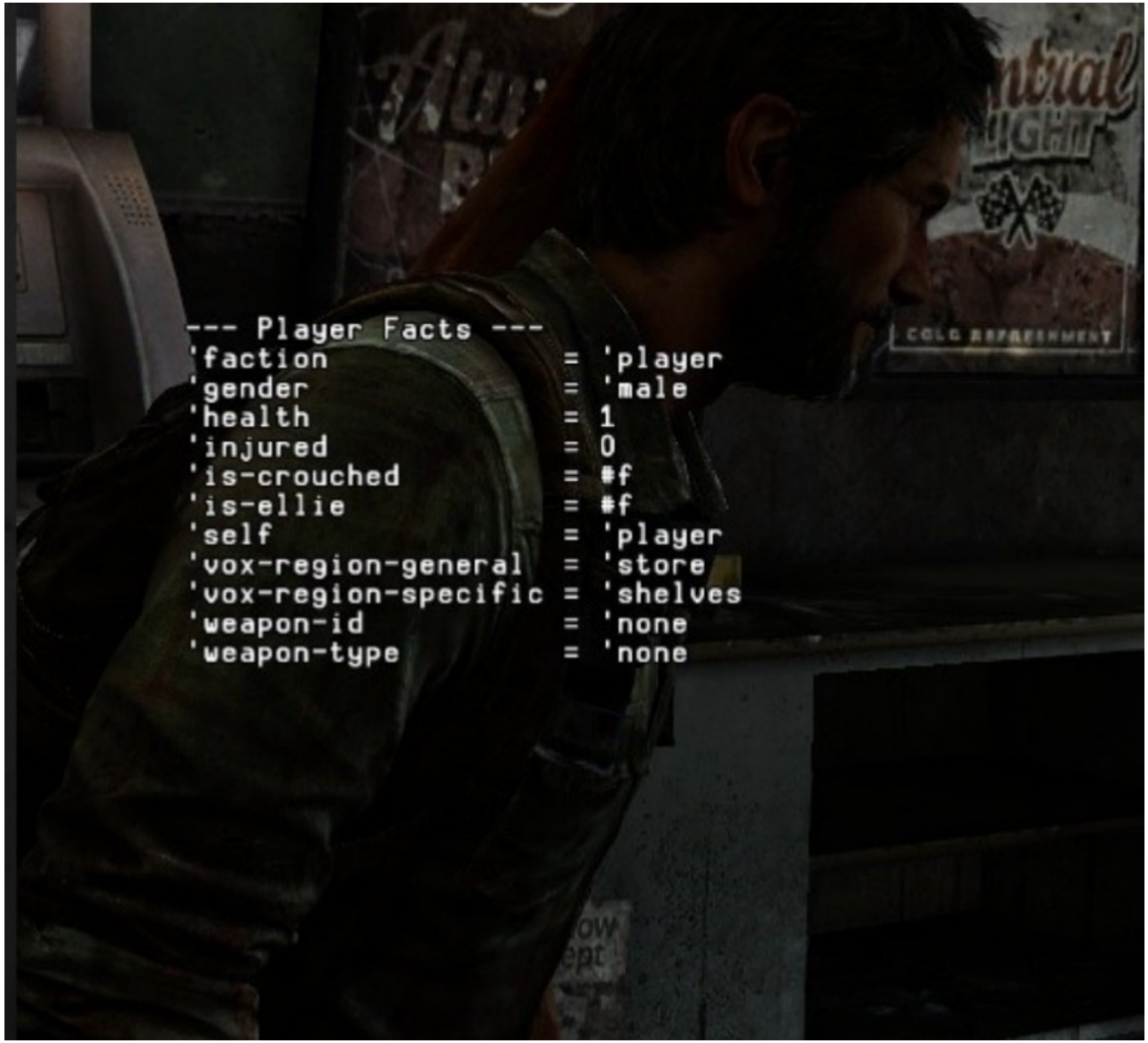
Branching Conversations



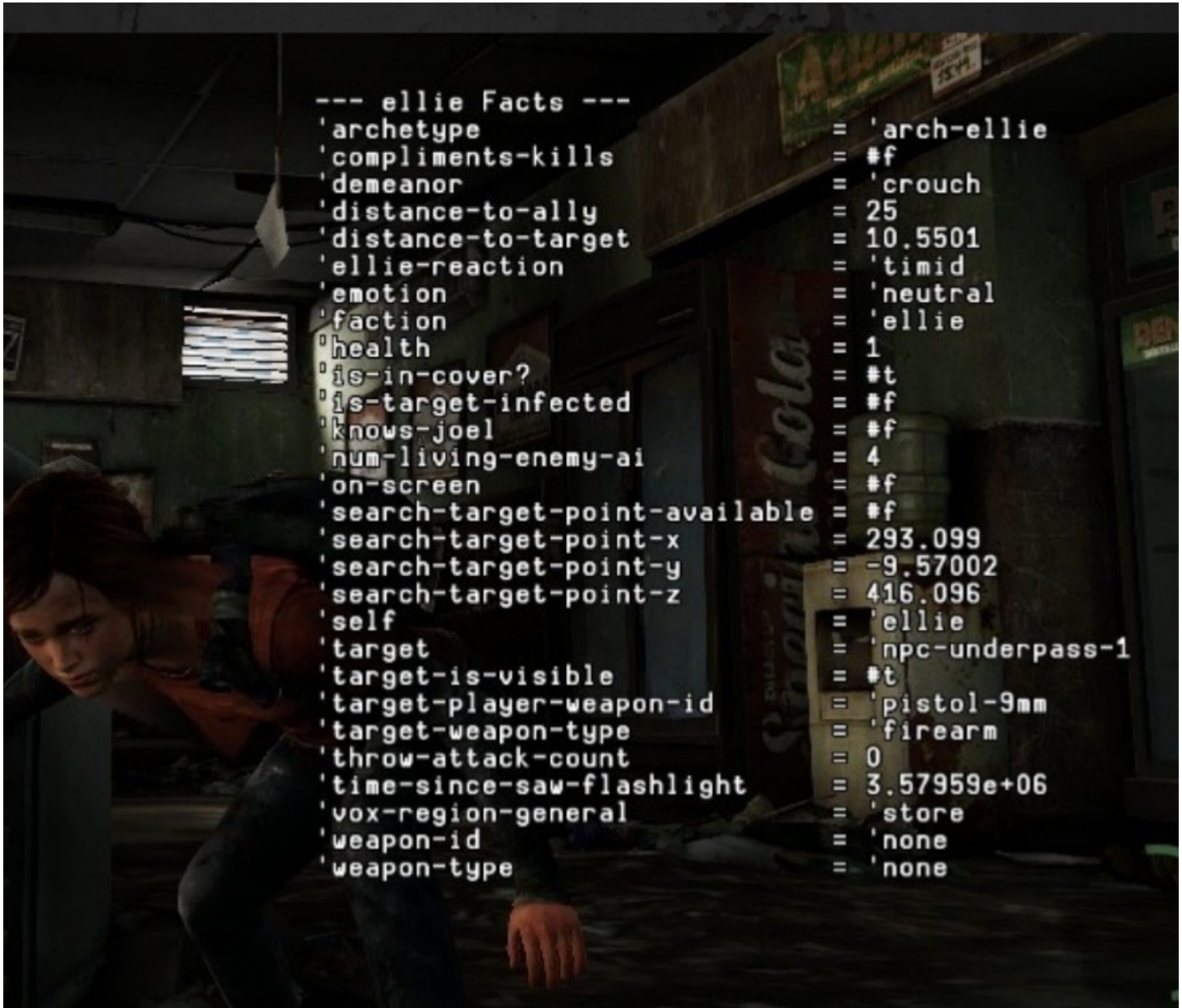
Fact Dictionaries



Joel’s Facts


--- Player Facts ---
'faction = 'player
'gender = 'male
'health = 1
'injured = 0
'is-crouched = #f
'is-ellie = #f
'self = 'player
'vox-region-general = 'store
'vox-region-specific = 'shelves
'weapon-id = 'none
'weapon-type = 'none

Ellie’s Facts

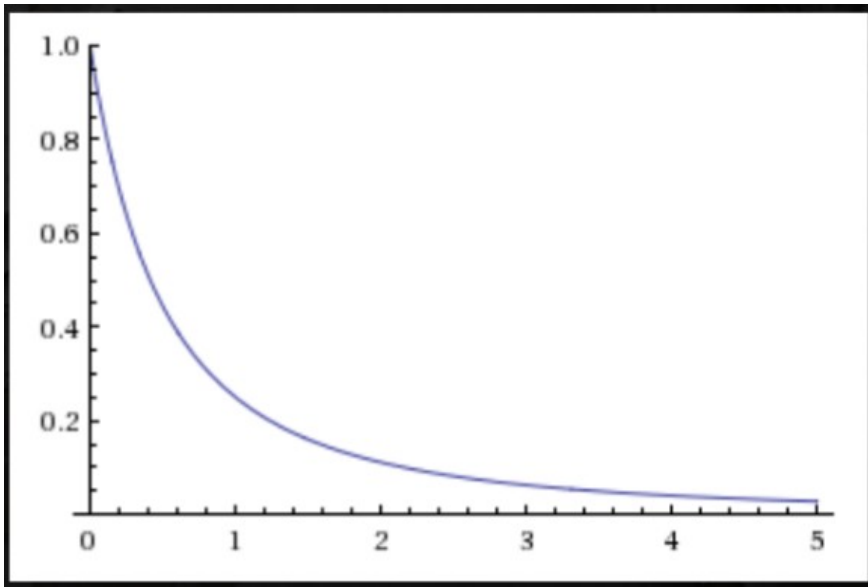


Adding Realism

Fall-Off

- Physically speaking, sound intensity follows an inverse square law

  $I(r) \propto 1/r2$
- This makes dialog difficult to understand when characters aren’t near the camera



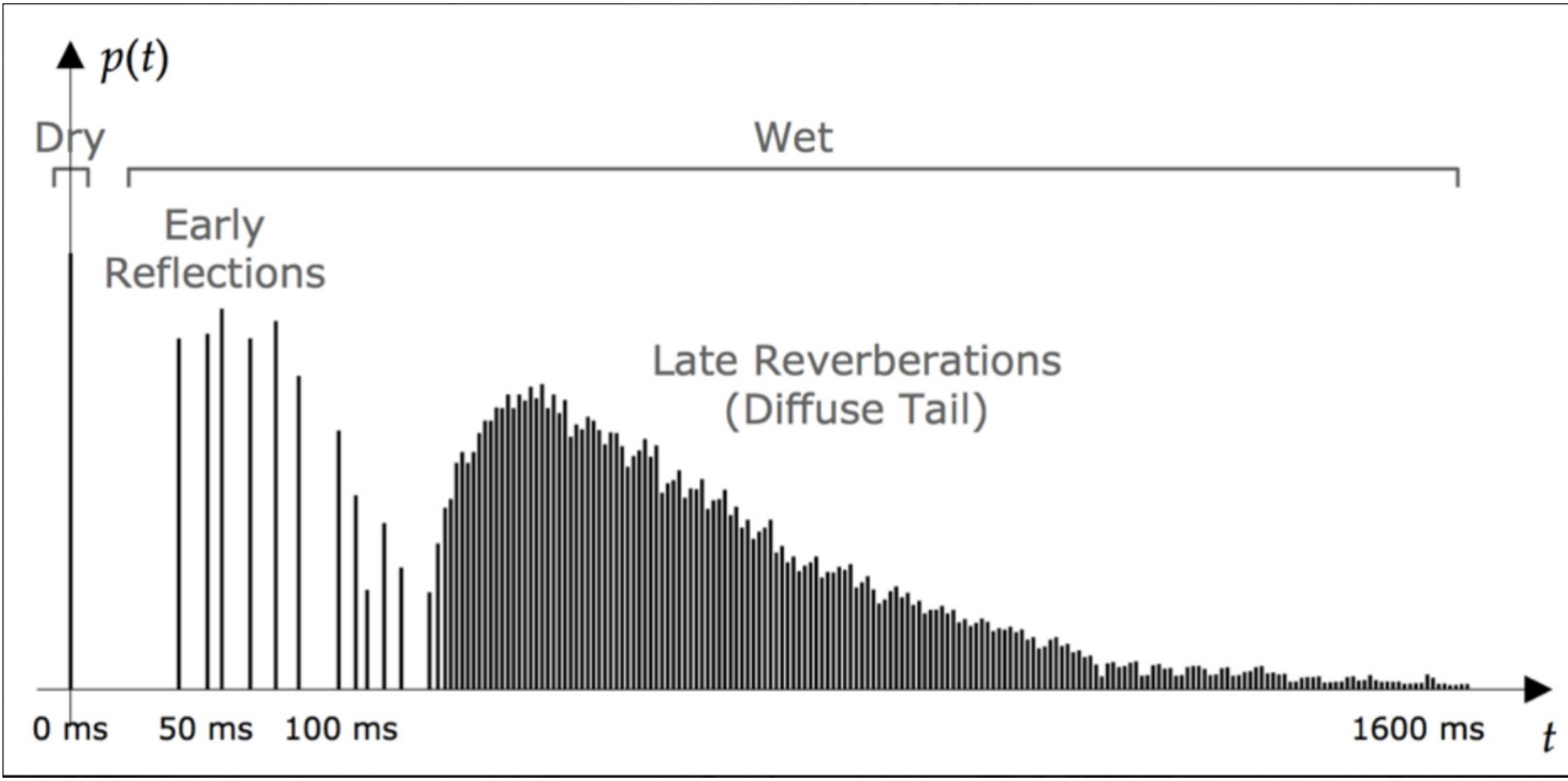
Fall-Off

- On The Last of Us, a “fake” fall-off curve for dialog (only) is used
  - Falls off slowly near the listener…
  - … more quickly in the mid-range…
  - … and slowly again in the far range

Reverb

- Two paths from sound source to the ear/microphone:
  - Direct path
  - Indirect path
- Gives rise to three sound qualities:
  - Dry (direct path)
  - Echo (early reflections) } “Wet”
  - Tail (late reverberations) } “Wet”

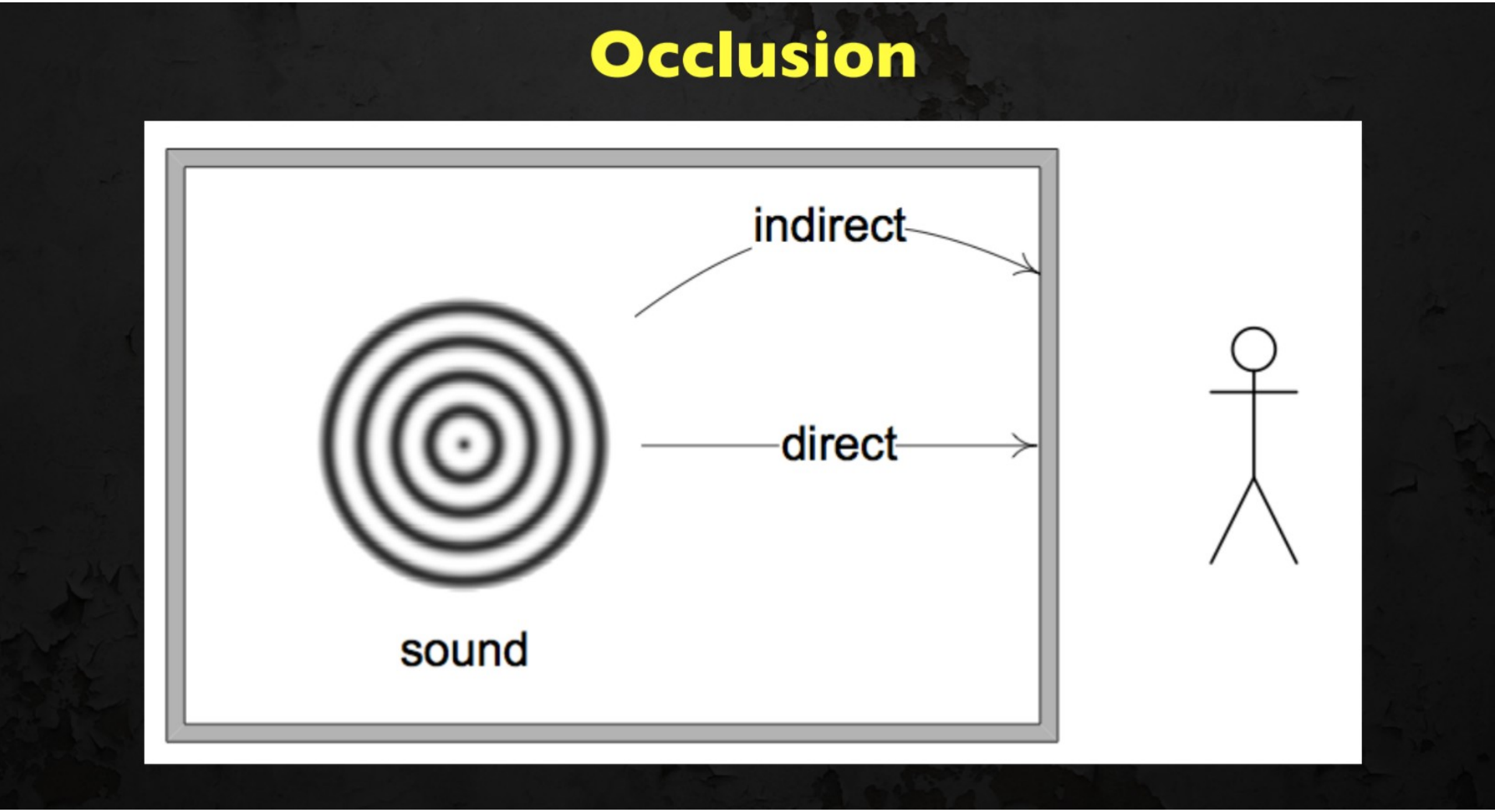
Reverb


p(t)
Dry
Wet
Early
Reflections
Late Reverberations
(Diffuse Tail)
0 ms
50 ms
100 ms
1600 ms
t

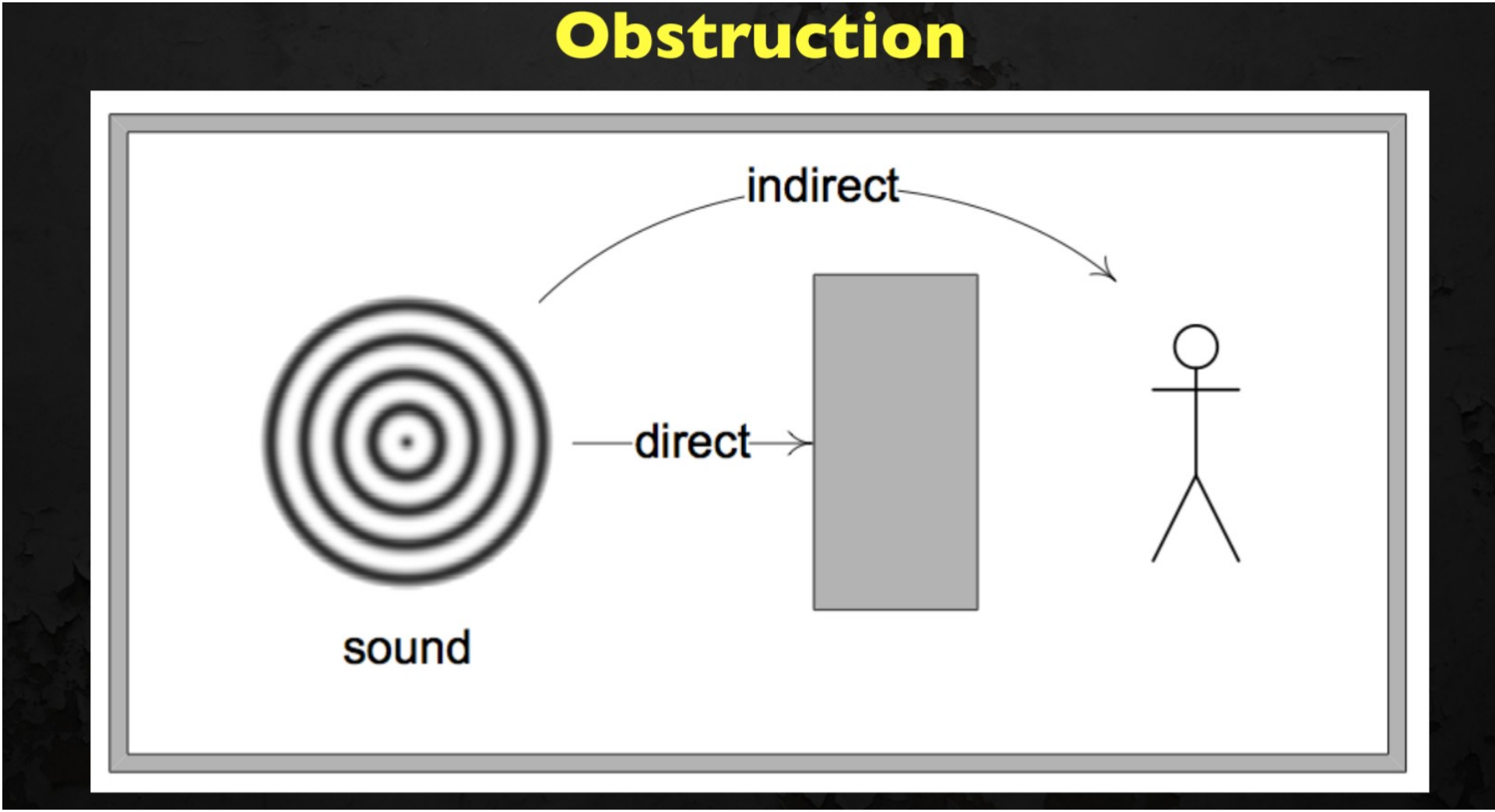
Audio Environments

- Regions placed by sound designers define the base reverb for different areas of the game world
  - reverb = characteristics of “wet” sound
  - provides cues about size of room and surrounding materials
- Reverb params blended across region boundaries

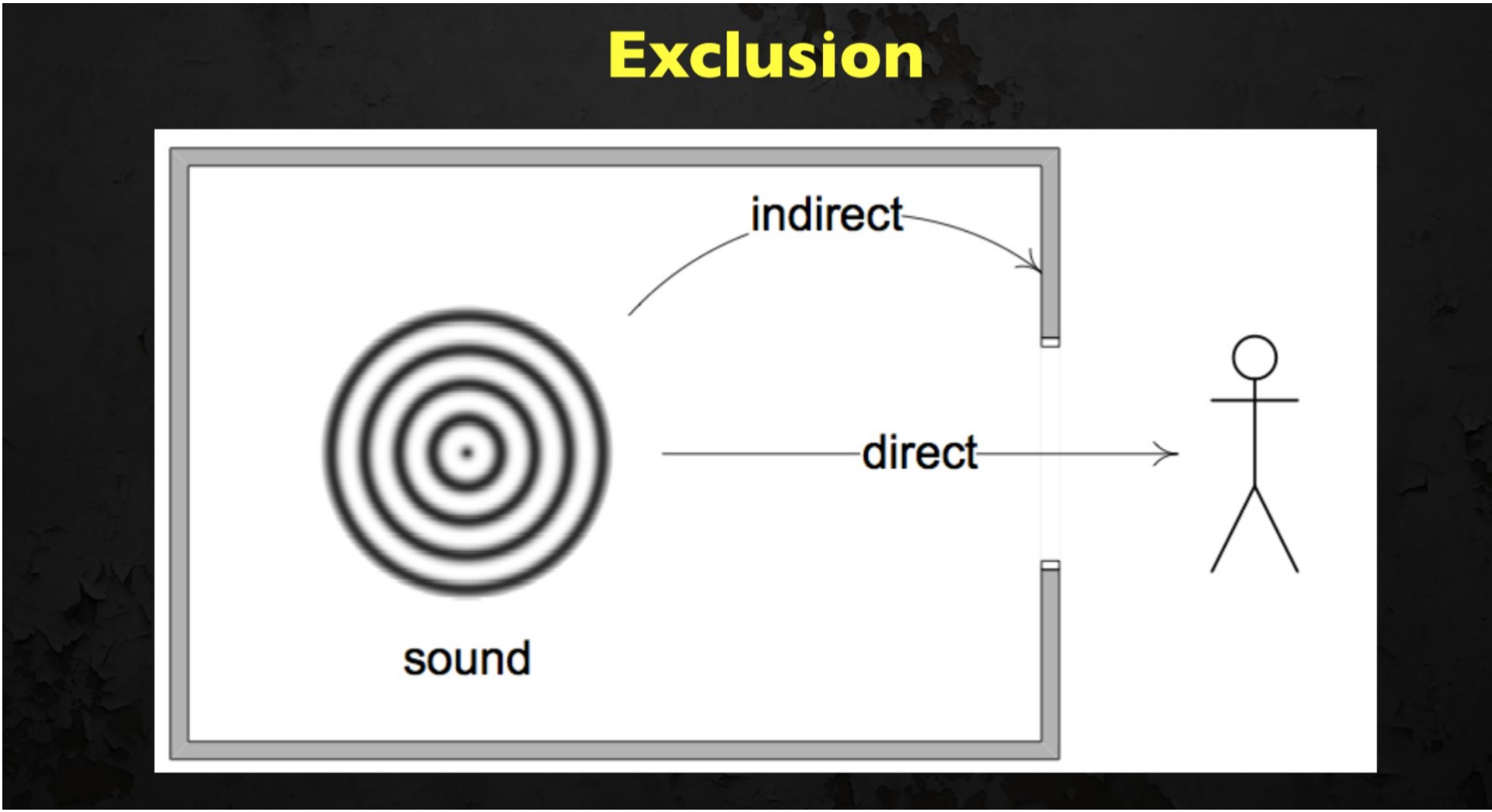
Occlusion, Obstruction and Exclusion



Occlusion, Obstruction and Exclusion



Occlusion, Obstruction and Exclusion



Final Notes

- you can get a lot of bang for your buck by investing in the animations and gestures
- directional gesture system -- allowing somewhat generic gesture
- animations to be played additively on top of a wide variety of full-body base anims
  - like walking or idling

Final Notes

- Just one example:
  - in action… pay special attention to Ellie’s friend Riley -- notice how her gestures are often directed TOWARD Ellie no matter where she might be standing


Joke
Quit

Dialogue Systems in Unity

- How they work • How to build them • Branching dialogue
- Based on the Game Dev Beginner tutorial
- gamedevbeginner.com/dialogue-systems-in-unity

Overview

- Dialogue system concepts
- Writing & storing dialogue
- Dialogue UI
- Dialogue logic
- Branching dialogue

Core Components

- Authoring (writing/storing text)
- Dialogue UI (on-screen text)
- The logic that connects them to gameplay

Simple Script-Based Dialogue

- Dialogue stored directly in a MonoBehaviour script.
- You need an approach to write and store dialogue that’s maintainable.

Script-Based Dialogue Code

- `[TextArea]`
- `public string[] lines;`
- this gets hard to manage in larger projects.

Problems With Script Storage

- Hard to reuse
- Tied to scene objects
- Difficult to scale

Using Scriptable Objects

- Scriptable Objects store dialogue separately
- from game logic and scenes.

Dialogue ScriptableObject Code

```
[CreateAssetMenu]
public class DialogueAsset : ScriptableObject
{
    [TextArea]
    public string[] dialogue;
}
```

Create these from Unity's Create → Dialogue Asset menu.

NPC Dialogue Reference

- NPCs store a reference to a DialogueAsset.

NPC Script Example

```
public class NPC : MonoBehaviour
{
    public string npcName;
    public DialogueAsset
dialogue;
}
```

Dialogue UI Components

- Panel GameObject
- Name Text (TMP)
- Dialogue Text (TMP)

Displaying Dialogue Code

```
dialoguePanel.SetActive(true);
nameText.text = npcName;
dialogueText.text = currentLine;
```

Dialogue Controller Purpose

- Controls dialogue flow, input,
- and UI updates.

Dialogue Controller Example

```
public void
StartDialogue(DialogueAsset
dialogue)
{
    currentIndex = 0;
    ShowLine();
}
```

Advancing Dialogue Code

```
public void NextLine()
{
    currentIndex++;
    ShowLine();
}
```

Ending Dialogue

- dialoguePanel.SetActive(false);

Branching Dialogue

- Dialogue choices allow different outcomes
- based on player decisions.

Dialogue Tree Data Example

```
[System.Serializable]
public class DialogueNode
{
    public string text;
    public DialogueNode[]
nextNodes;
}
```

Handling Player Choices

- UI buttons trigger selection
- and load the next dialogue node.

Next Steps

- Add typewriter effect
- Add dialogue choices UI
- Connect dialogue to quests
- Consider full dialogue tools