

Lecture 21

Level Design

Do We Really Need Level Design?

- Level design makes sense for single player games
- What if our game is **open world**?
 - Each location is a level
 - All that changes is the transition
- What if our game is **multiplayer**?
 - Are the maps always the same?
 - What about game modes?
- What if is a **strategic card game** (e.g. *Magic*)?
 - Are all the cards available at start?
 - How does someone learn how to play?

What is Level Design?

- Layout of **game geography**
 - Location and relationship of challenges
 - Movement of dynamic features (e.g. NPCs)
- Understanding of **player capabilities**
 - Abilities, mechanics available to the player
 - Assumptions of current player skill level
- Layout of **player progression**
 - How the player should move through the game
 - How the player visualizes this progression

Aspects of Game Design

- Games as **Exploration**
 - Focuses on game *geography* and *capabilities*
 - Typically involves heavy storyboarding
- Games as **Education**
 - Train player skill and understanding
 - Focuses primarily on *player capabilities*
- Games as **Storytelling**
 - Focuses on *player progression*
 - Most challenging element of game design

Aspects of Game Design

- Games as **Exploration**
 - Focuses on game *geography* and *capabilities*
 - Typically involves heavy storyboarding
- Games as **Education**
 - Train player skill and understanding
 - Focuses primarily on *player capabilities*
- Games as **Storytelling**
 - Focuses on *player capabilities*
 - Most important element of game design

Not in this Lecture

Aspects of Game Design

- **Games as Exploration**

- Focuses on game *geography* and *capabilities*
- Typically involves heavy storyboarding

- **Games as Education**

- Train player skill and understanding
- Focuses primarily on *player capabilities*

- **Games as Storytelling**

- Focuses on *player capabilities*
- Most important element of game design

Not in this Lecture

Players Want to Explore the World

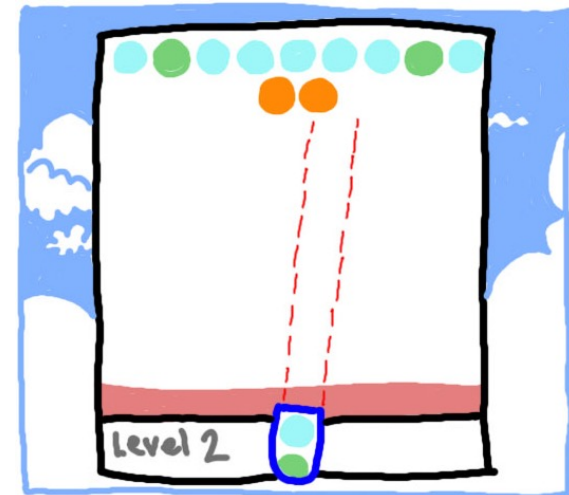
- Exploring the **physical space**
 - What happens when I go here?
 - **Example:** Any open world RPG
 - But does not require complex game world
- Exploring the **ludic space**
 - What happens when do this action?
 - Requires deep, complex interactions
 - **Example:** Goofing on Bethesda NPCs

Storyboarding

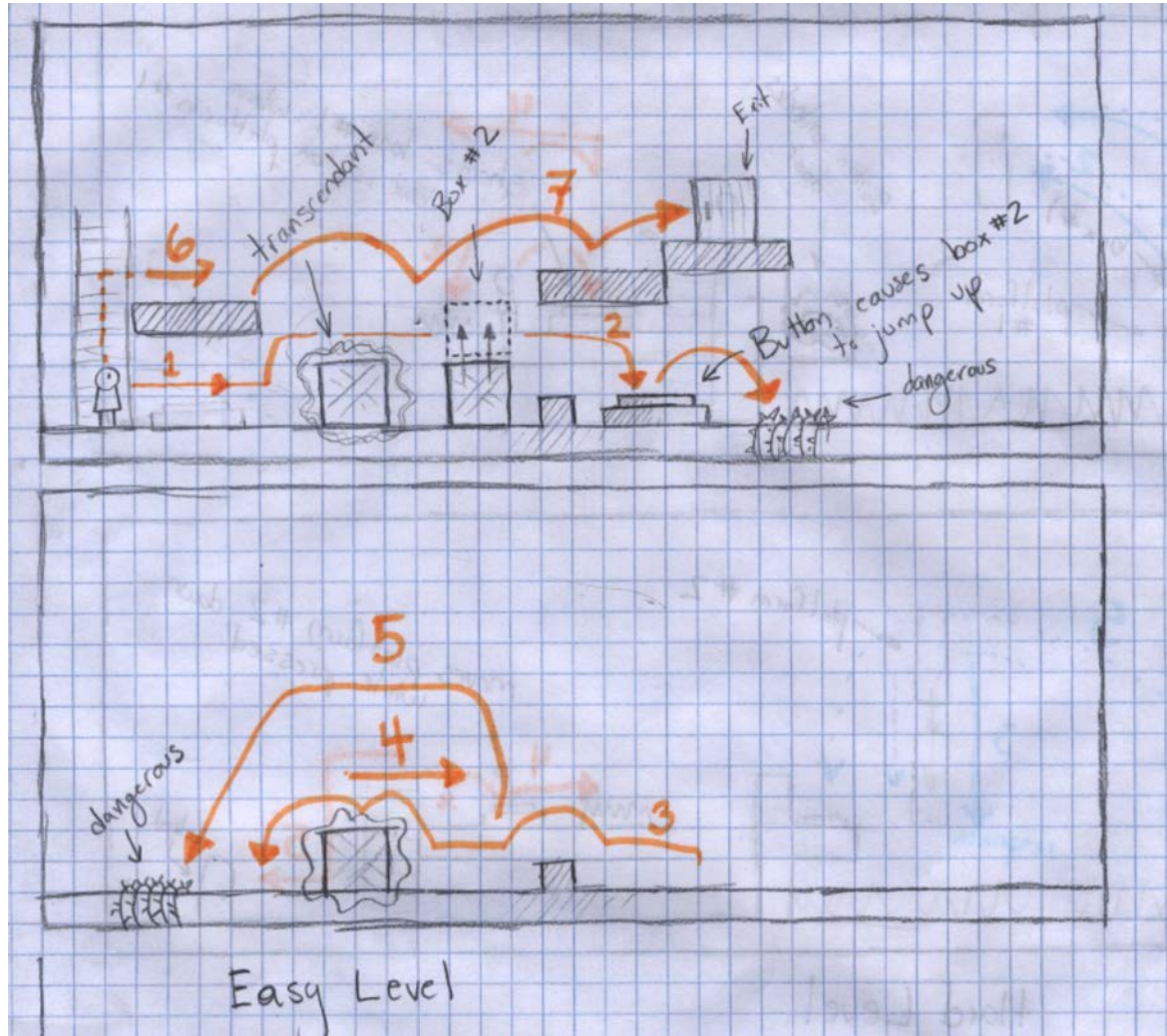
- Diagrams player action throughout level
 - Different from film storyboarding
 - Currently a bunch of *informal practices*
- **Disembodied Action**
 - Action corresponding to UI elements
 - **Example:** Buttons, menus
- **Embodied Action**
 - Action that is tied to a character/avatar
 - Typically maps player movement in level

Disembodied Action: Cause and Effect

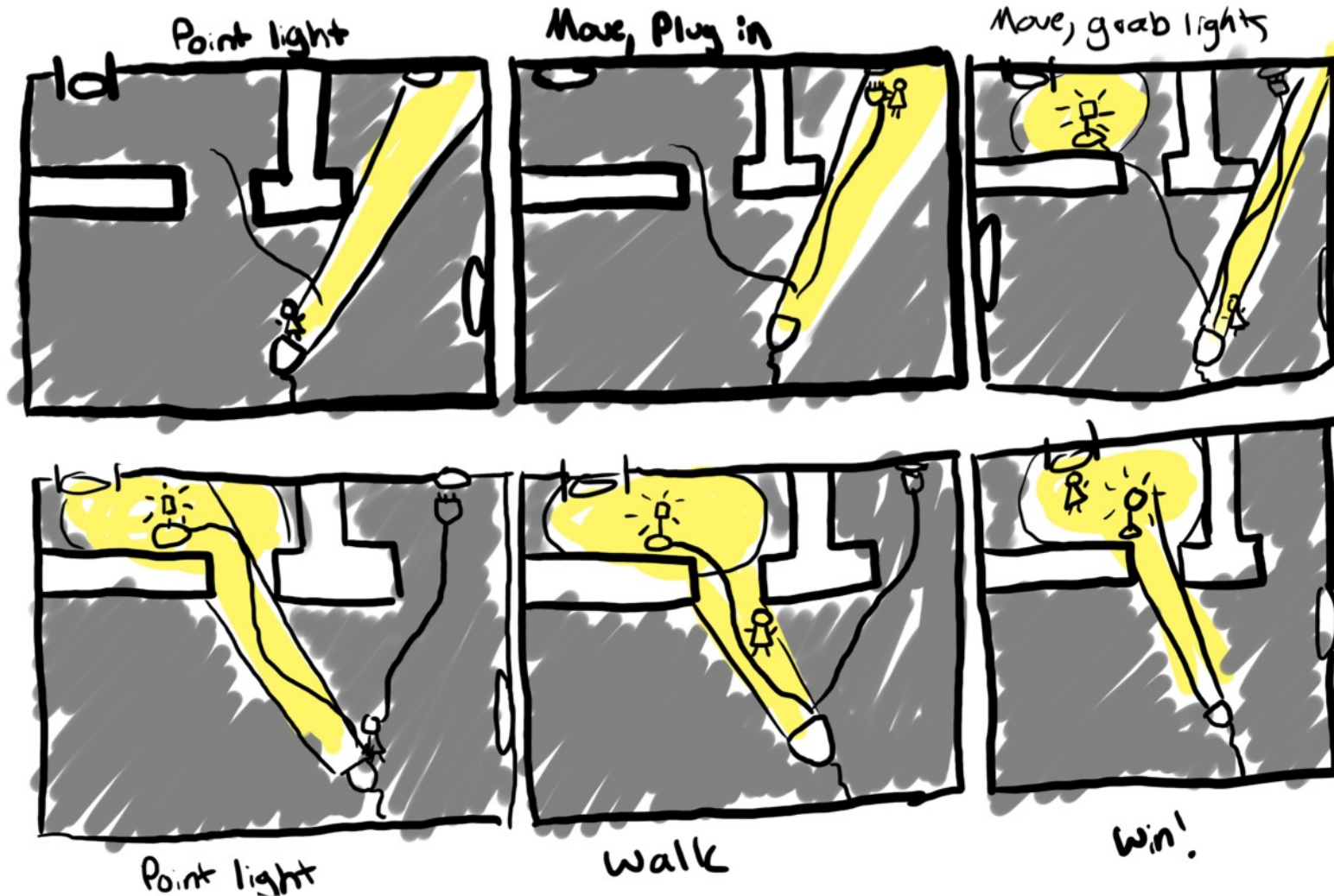
- **Draw the initial scene**
 - Could be the entire level
 - Zoomed in portion of screen
 - Must capture area that will be affected by the action
- **Indicate the action**
 - Draw mouse pointer
 - Indicate gamepad button
 - Annotate with a “tool tip”
- **Draw the action effect**
 - Change in initial scene



Embodied Action: Single Scene



Embodied Action: Multiple Scenes

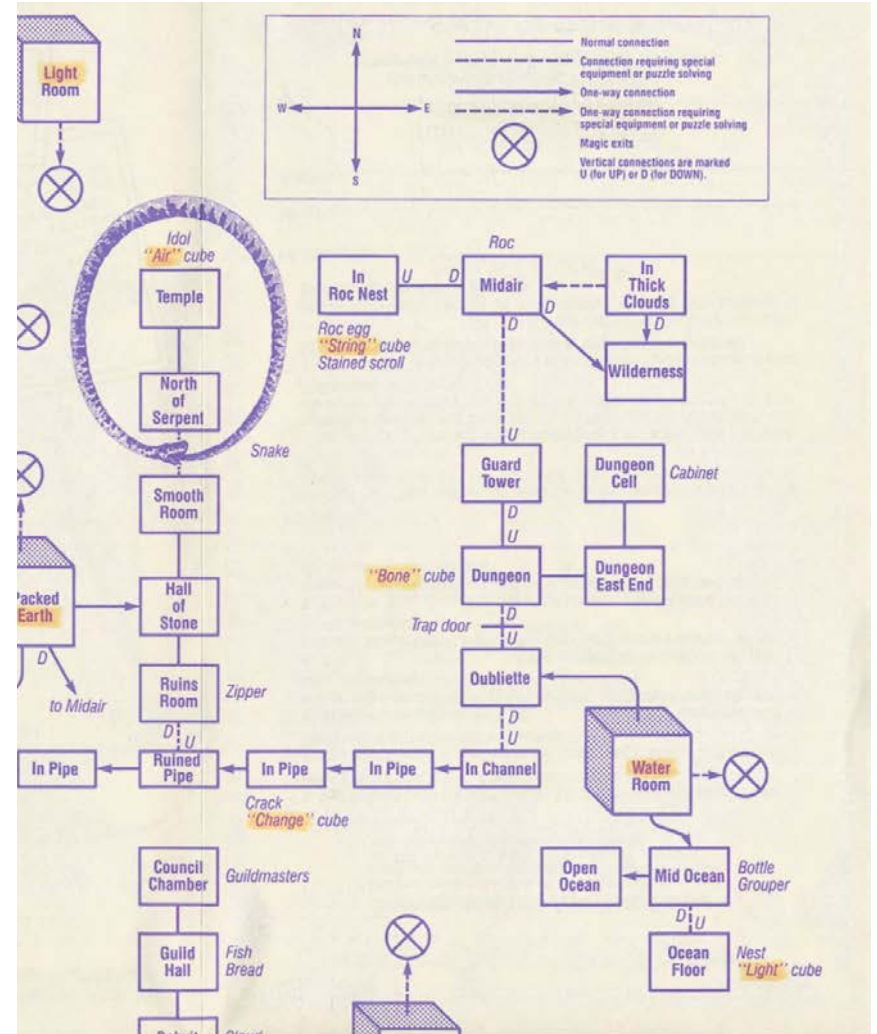


But There is a Problem

- You are **not** the player!
 - You storyboard what you *think* player will do
 - Player may do something completely *different!*
- Level design is about **constraining** player
 - You design level to force player to do things
 - Challenges are doors blocking progress
 - Player must use skill to open the door
- Storyboarding **maps** these constraints

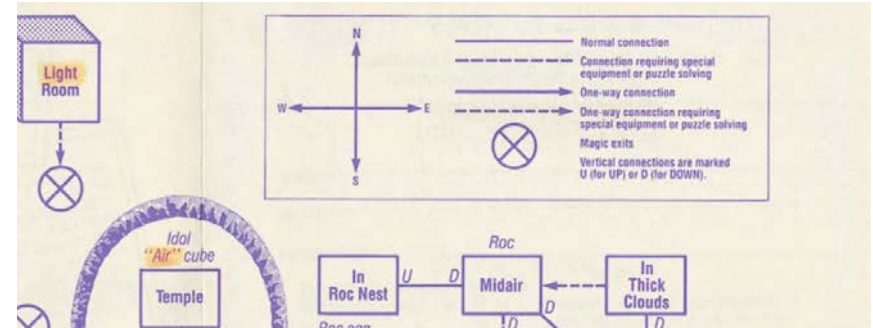
This is How it Ever Was

- Classic text adventures...
 - Goal is location to reach
 - Locked doors block progress
 - Use actions to unlock doors
- Still design in same way
 - Challenges block the goal
 - Use mechanics to overcome
- Design levels with...
 - **Discrete challenges** (doors)
 - Put together **intelligently**



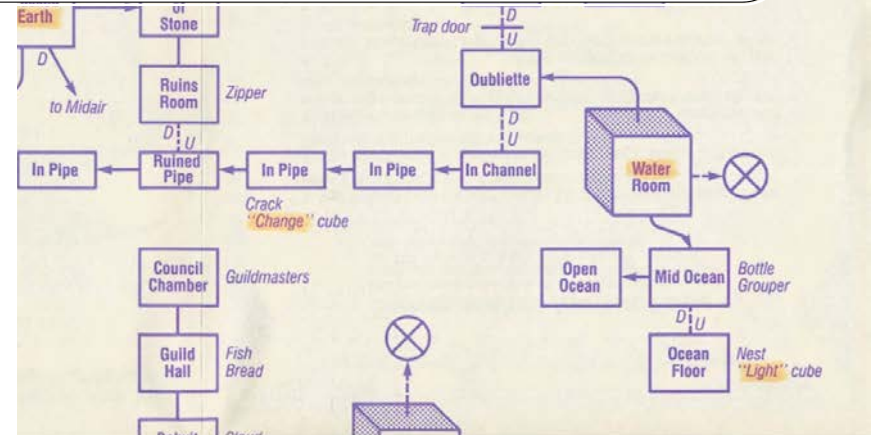
This is How it Ever Was

- Classic text adventures...
 - Goal is location to reach
 - Locked doors block progress



Tight Level Design = Tight Challenge Spacing

- Use mechanics to overcome
- Design levels with...
 - **Discrete challenges** (doors)
 - Put together **intelligently**



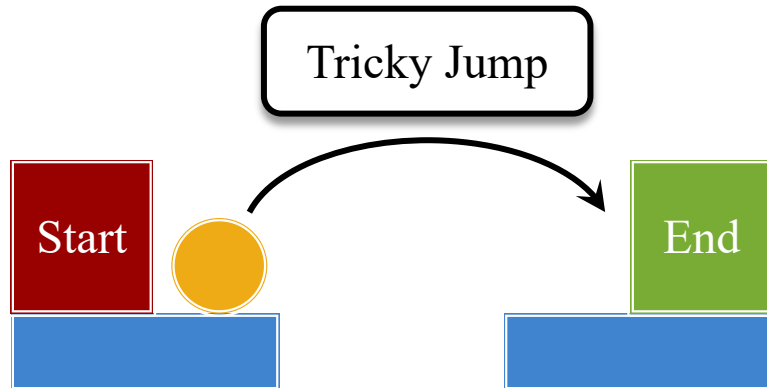
Design Patterns

- Design uses building blocks
 - Mechanic/challenge pairs
 - Start and end location
 - String together to make level
- Key building block features
 - Requires verb/interaction
 - Must be possible to *fail*
 - Difficulty is *tunable*
- **Patterns** are common blocks
 - Appear many times in game
 - Even across multiple games

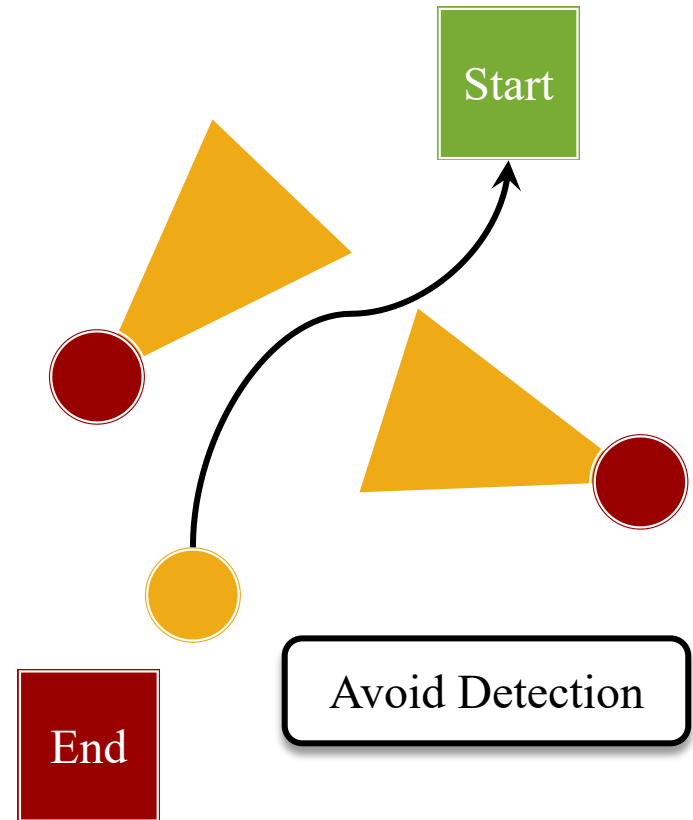


Design Pattern Examples

Platformer

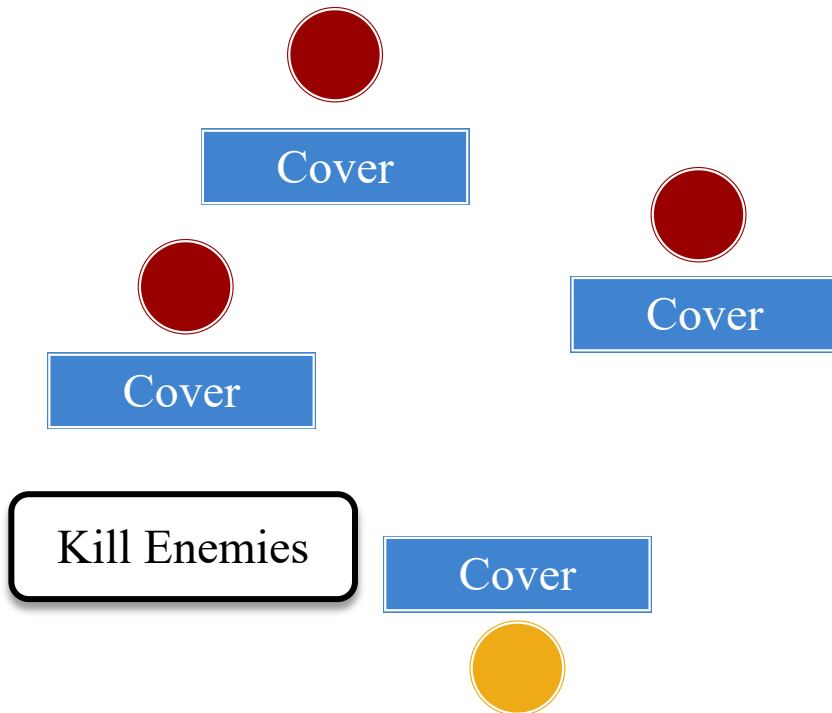


Stealth Game

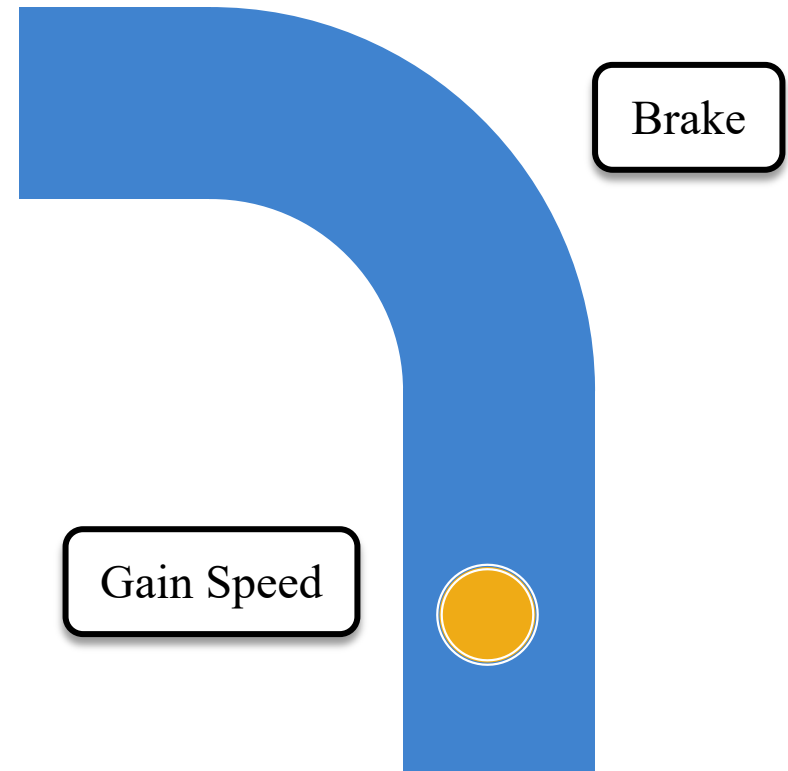


Design Pattern Examples

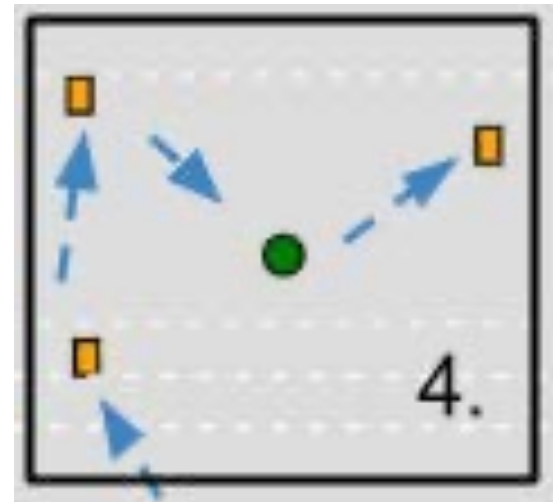
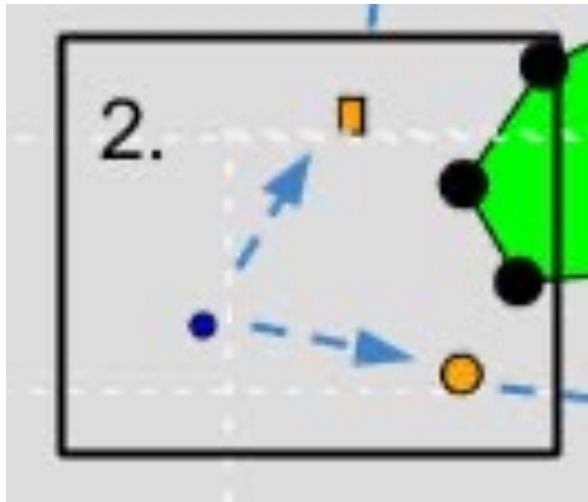
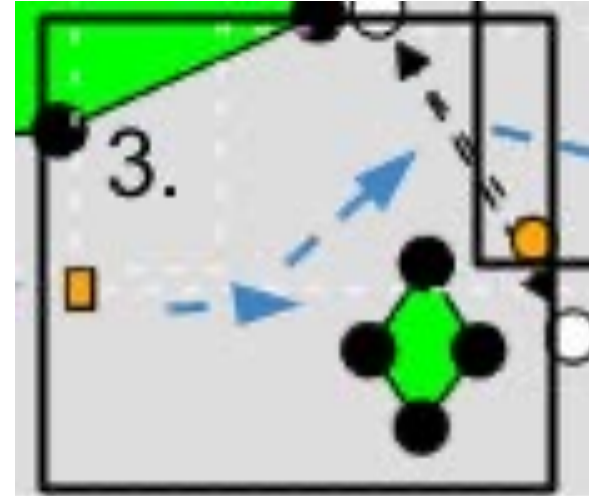
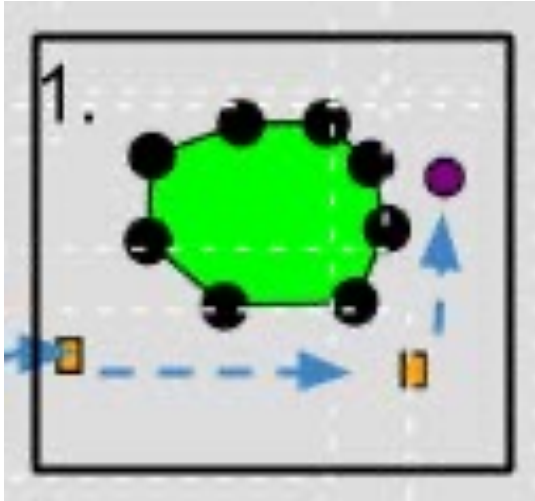
Shooter/Action Game



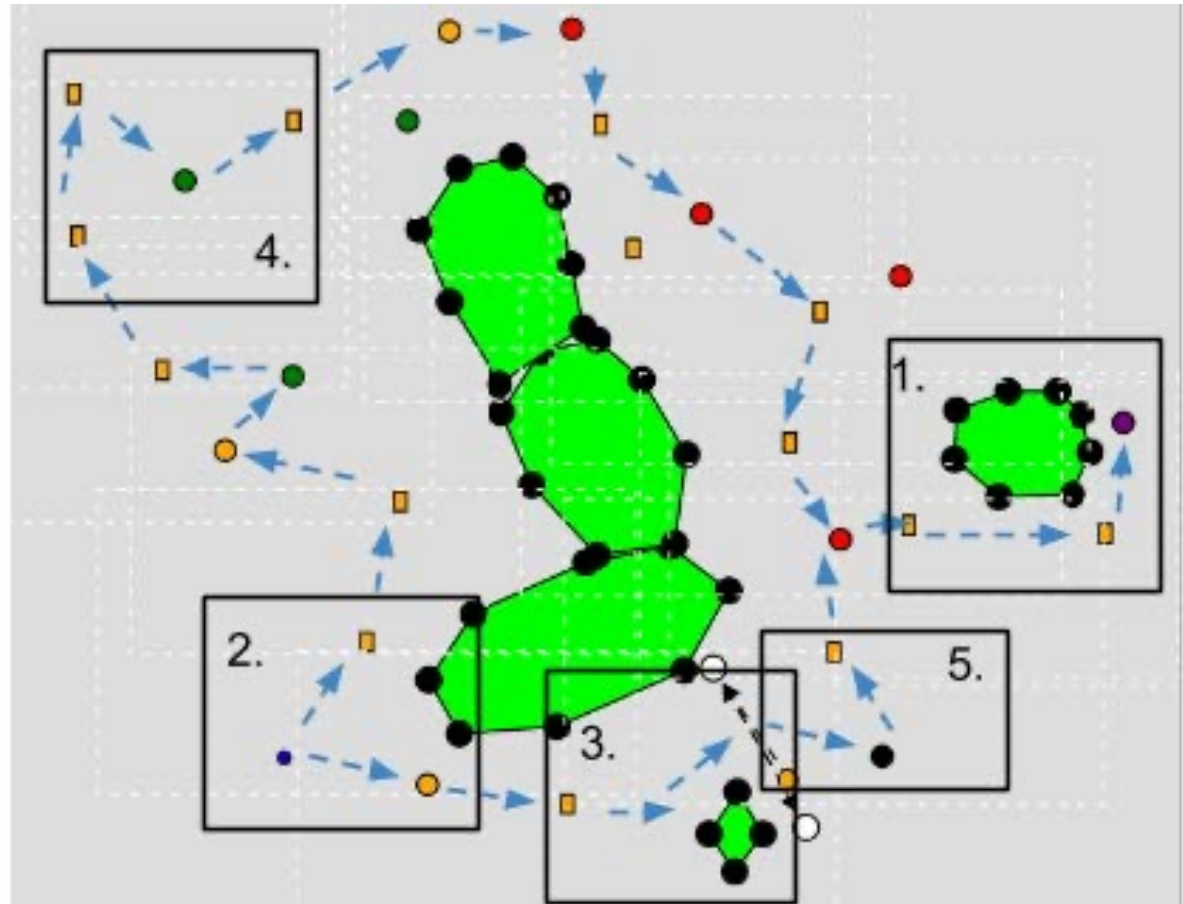
Racing Game



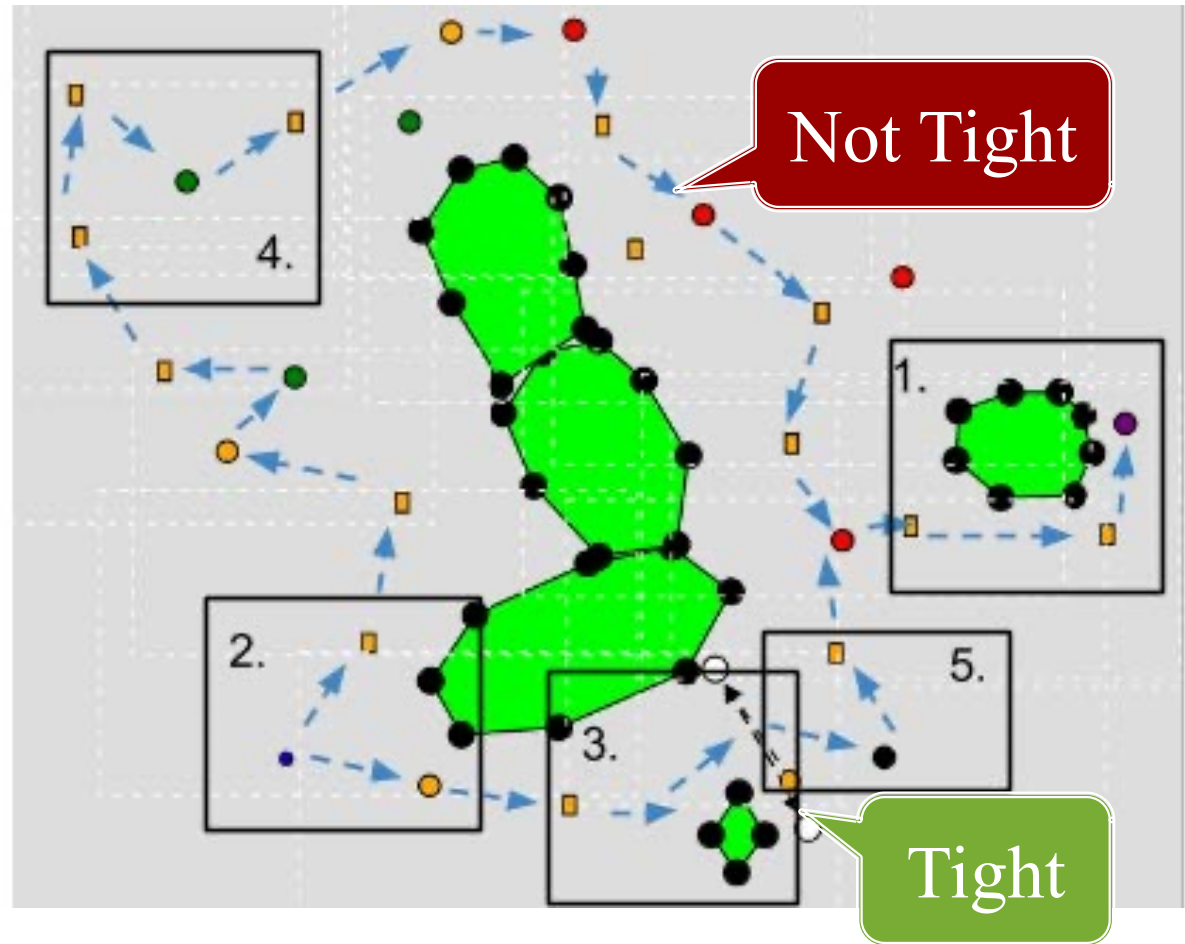
Dash: Basic Design Patterns



Dash: Putting it All Together



Dash: Putting it All Together

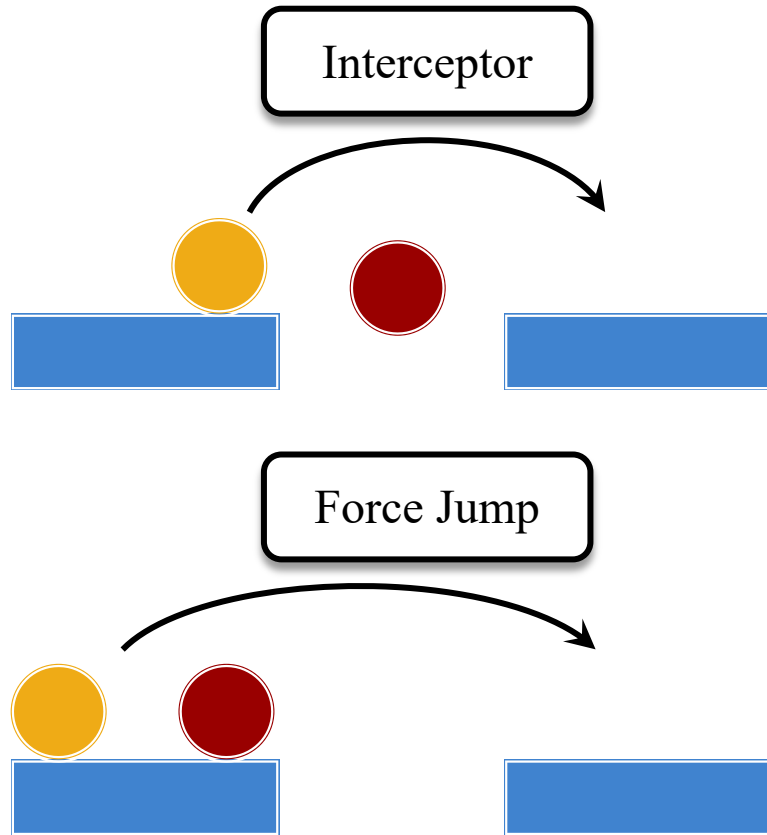


Composite Patterns

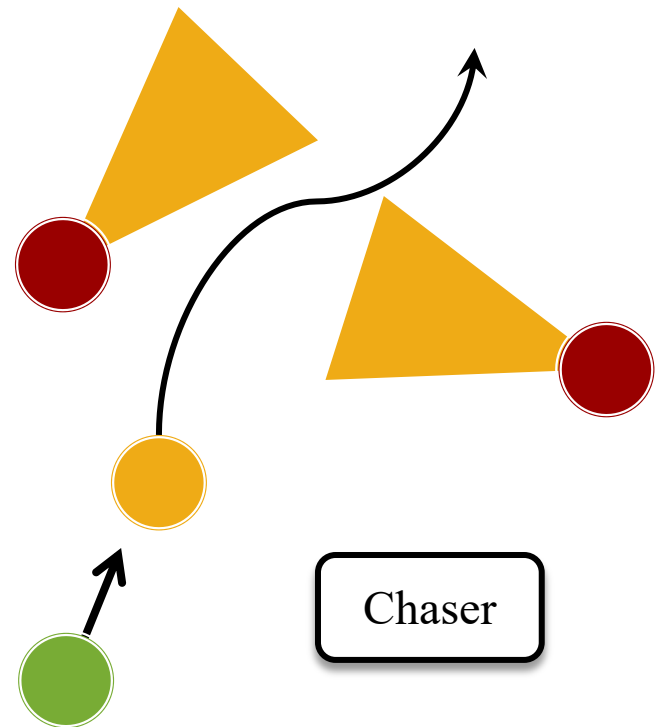
- Piecewise design creates a very linear feel
 - **Pattern A** followed by **Pattern B** followed by...
 - Player is explicitly aware of building blocks
- **Composite patterns** allow for variations
 - Two patterns combined in the same space
 - Makes original pattern much more difficult
 - Player now has to react to them both
- **Reading: Extended/Evolutionary Challenge**

Composite Patterns

Platformer

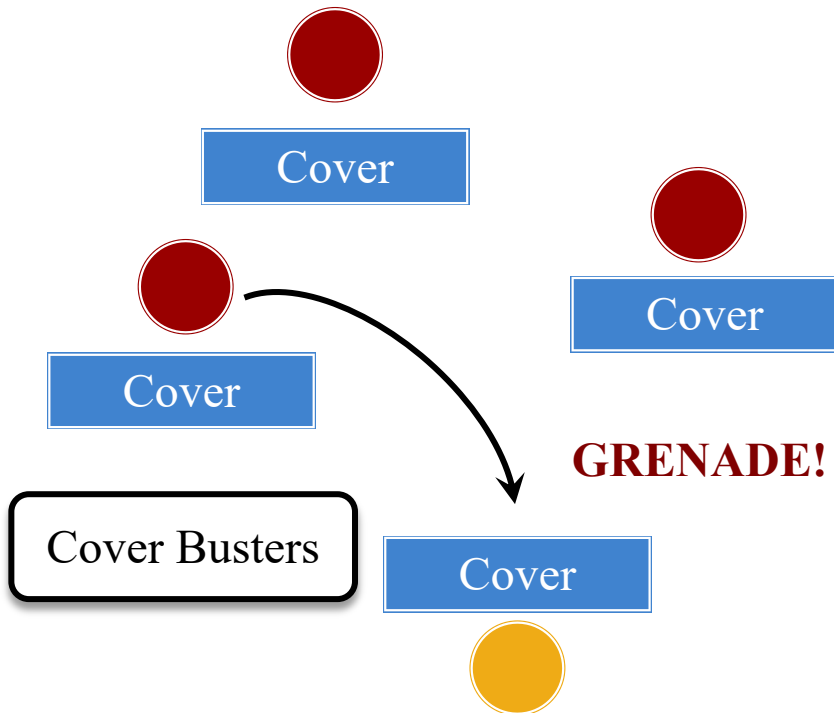


Stealth Game

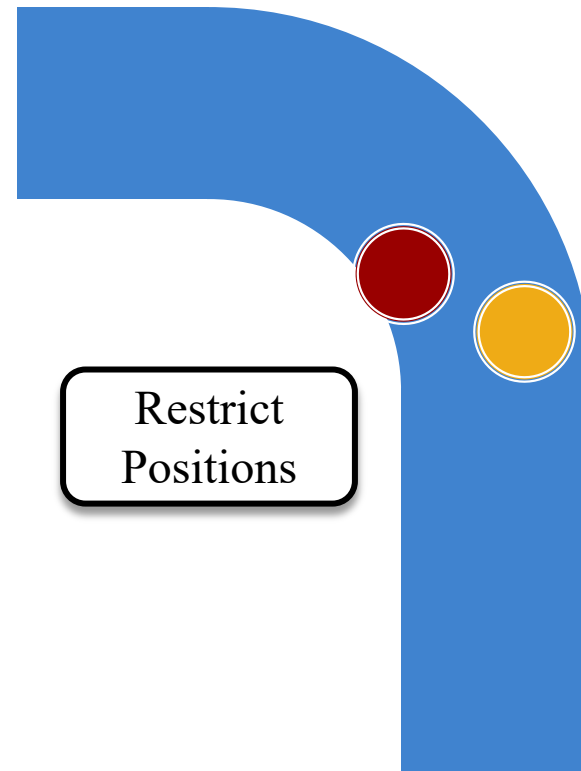


Composite Patterns

Shooter/Action Game



Racing Game



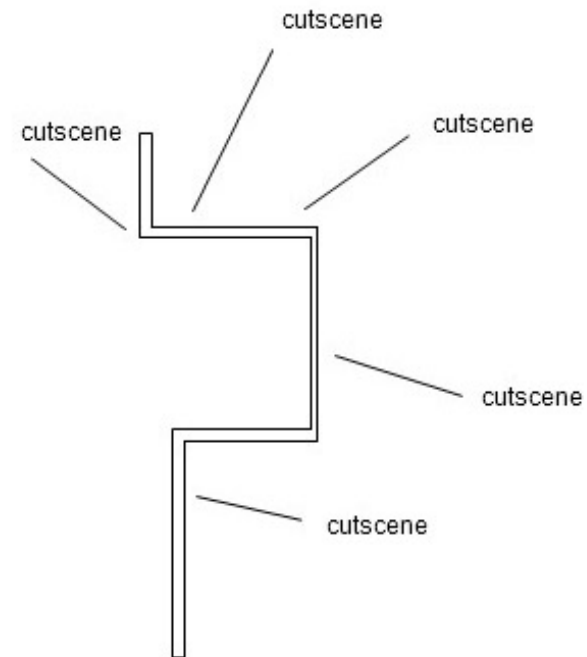
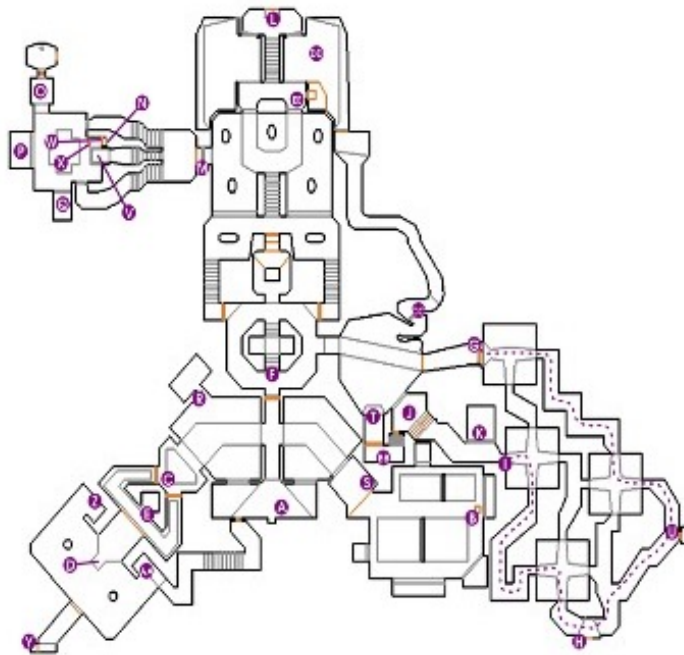
Is Linearity a Problem?

[Image attribution unknown]

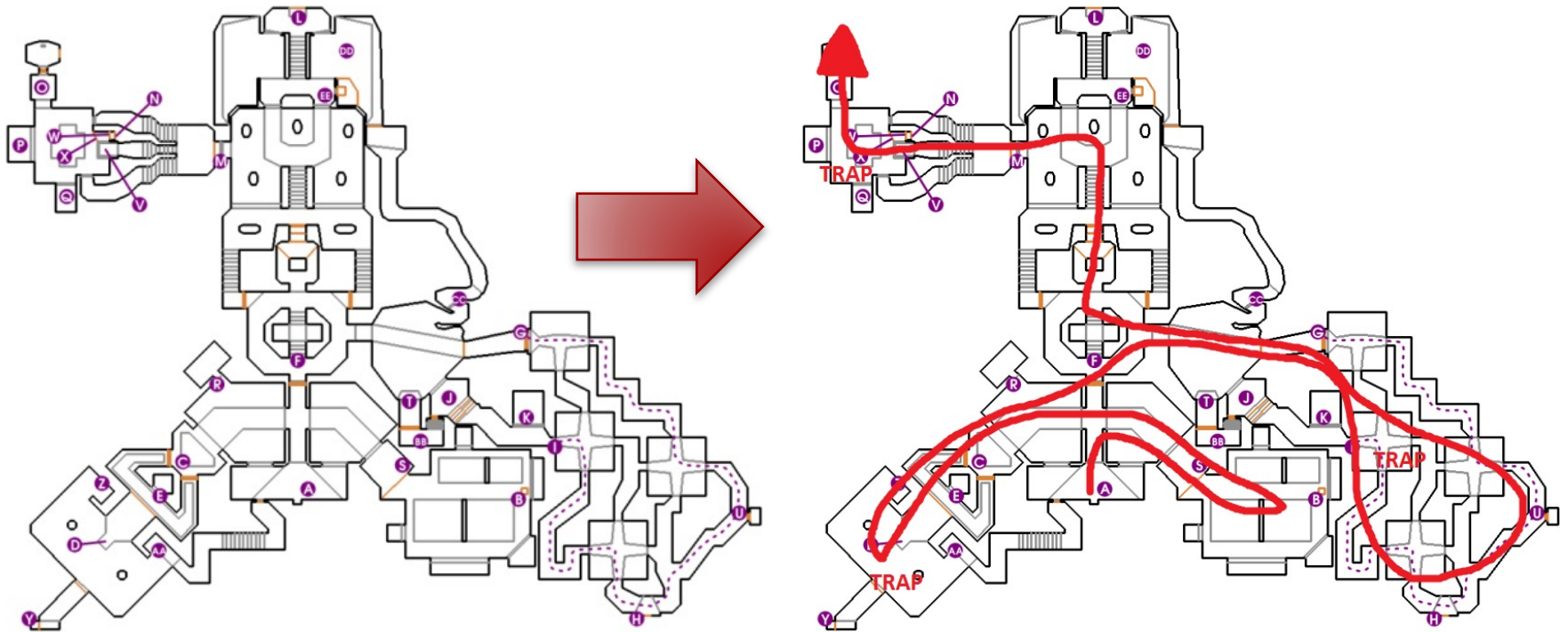
FPS map design

1993

2010

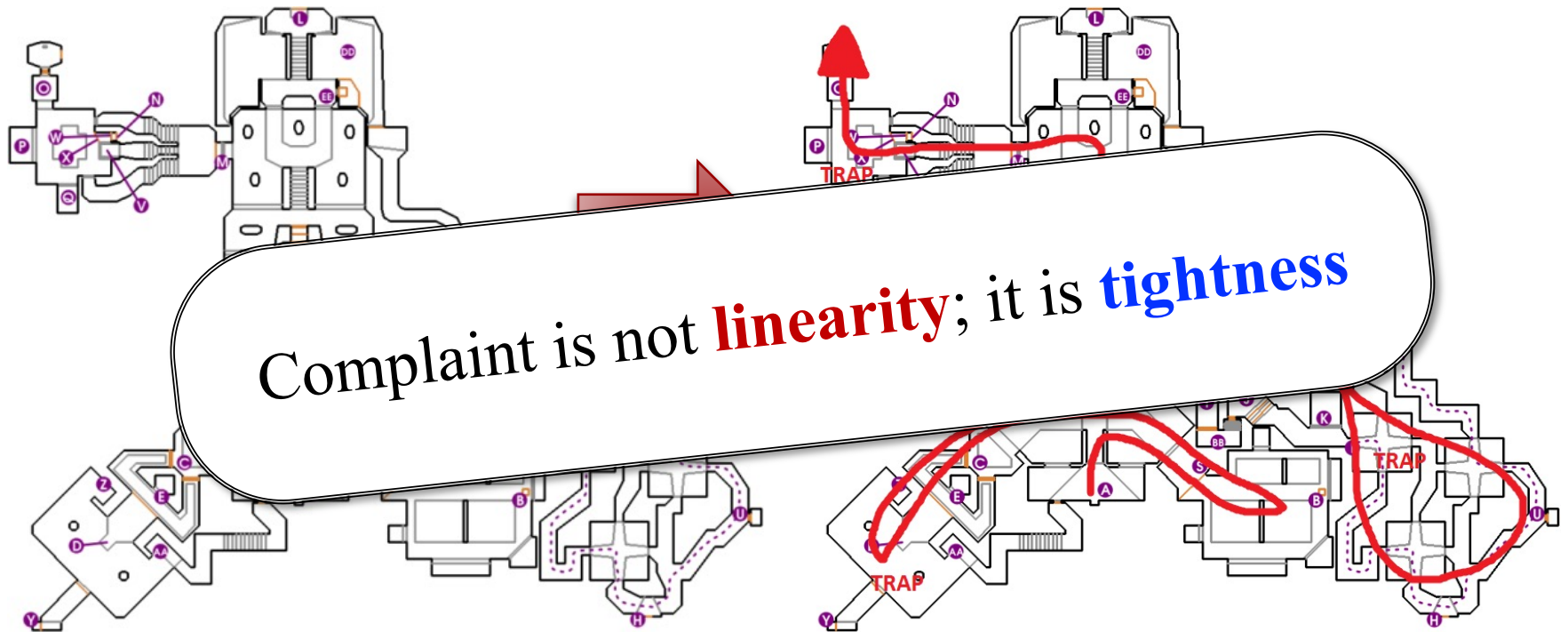


But Actually...



[refugeinaudacity.wordpress.com]

But Actually...



[refugeinaudacity.wordpress.com]

Aspects of Game Design

- Games as **Exploration**
 - Focuses on game *geography* and *capabilities*
 - Typically involves heavy storyboarding
- **Games as Education**
 - Train player skill and understanding
 - Focuses primarily on *player capabilities*
- Games as **Storytelling**
 - Focuses on *player progression*
 - Most challenging element of game design

Learning How to Play

- Mechanics are (often) new and unfamiliar
 - Players have to learn how to interact with them
 - **Aside:** why innovation is not always popular
- Players could learn by reading the *manual*
 - This is boring! Let me play already
- **Tutorial levels** allow the player to...
 - Get started playing immediately
 - Learn the mechanics while playing

Classic Approach: Restrict the Player

- Start with your **gameplay specification**
 - Remove all but the barest mechanics
 - Remove verbs by disabling controls
 - Remove interactions by omitting "board elements"
- Levels add new mechanics back one at a time
 - **Example:** Platformer with a "no-jump" level
- Do not need to add a new mechanic each level
 - "Deep" mechanics allow many levels per mechanic
 - This can influence game geography (e.g. worlds)

Example: Starcraft Campaign



Explicit Restrictions

- Mechanics are unavailable for current level
 - Controls for actions are explicitly disabled
 - Interactions disabled, even if elements present
- **Motivation:** Prevents player confusion
 - Do not waste time on useless mechanics
 - Key in the casual and young audience
- **Examples:** Many AAA commercial games
 - *Starcraft* single-player campaign
 - *Portal* (integrated into story)

Implicit Restrictions

- Mechanics are always available, but not needed
 - Challenges designed for an explicit mechanic
 - Other mechanics may succeed, but they are harder
 - Level has hints to guide player to right mechanic
- **Motivation:** Allow replay in tutorial levels
 - Players go back and try optional approaches
 - Achievements are structured to encourage this
- **Example:** Many amateur Flash games
 - *My First Quantum Translocator*

The Tyranny of Choice

- Too much choice can make us unhappy
 - We are often paralyzed by what to do
 - Studied by Myers & Lane; popularized by Barry Schwartz
- But games are about **meaningful choice**
 - Problem is when choices are too similar
 - Good choices must be *significantly* different
 - **Example:** Dagger adds +1 bonus to a stat of 102
- Players use rough heuristics for making choices
 - Pattern match current situation to determine action

The Tyranny of Choice

- Too much choice can make us unhappy
 - We are often paralyzed by what to do
 - Studied by Myers & Lane; popularized by Barry Schwartz
- But choice is also important
 - **Limiting choice helps train player**
 - Good choices are often limited
 - **Example:** Dagger adds +1 bonus to a stat of 102
- Players use rough heuristics for making choices
 - Pattern match current situation to determine action

Portal 2 Mechanics



Mechanics

● Introduction

● Variation

Grab

Grab with Portal Device

Grab in flight

Portals

Single Portal Device

Dual Portal Device

Weighted Storage Cube

Multiple cubes

Heavy Duty Super Colliding Super Button

Multiple buttons

Emancipation Grill

Timed switches

Switches

Panels

Momentum

Toxic Water

Thermal Discouragement Beam

Multiple beams

Discouragement Redirection Cube

Unstationary Scaffold or Victory lift

New Mechanics

Recombination

Kill inside chamb

Reinforcement

How long to “dwell” on mechanic before a new one?

Actions:

A = jump

B = dash

A B

vs.

A A A B

Recombination

How often to combine with other mechanics

Actions:

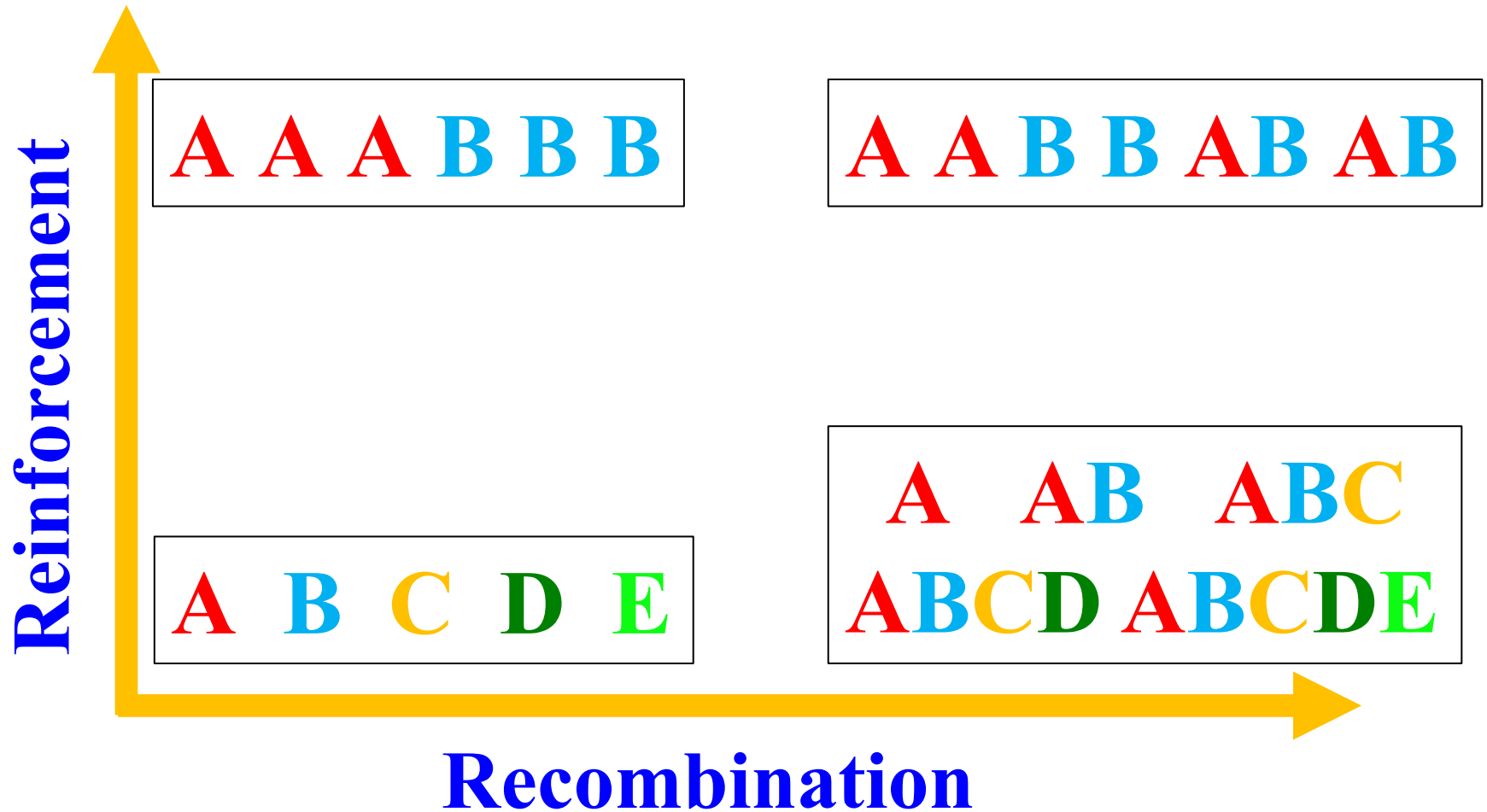
A = jump

B = dash

C = shoot fireball

A B C vs. **A AB ABC**

Reinforcement vs. Recombination



Robot Unicorn Attack



Robot Unicorn Attack Progression

Mechanics:

A = jump

B = dash

A A A B A A B

High reinforcement, low recombination

Hello World!



COMBO(C) REWIND(V)
QUIT RESTART(R)

TIME: 7
PAR TIME: 45
SPEED TIME: 12

ALPHA

COINS 🟡: 1/6
STARS ⭐: 0
POINTS: 0

Hello World!

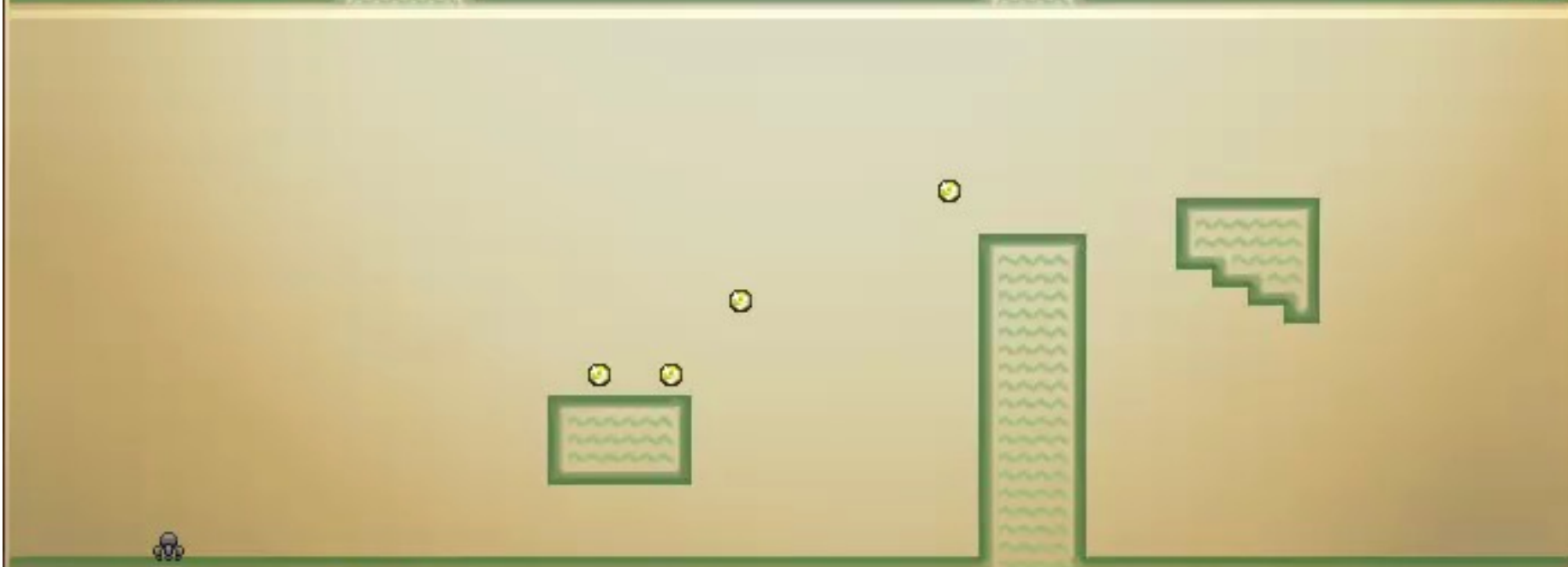
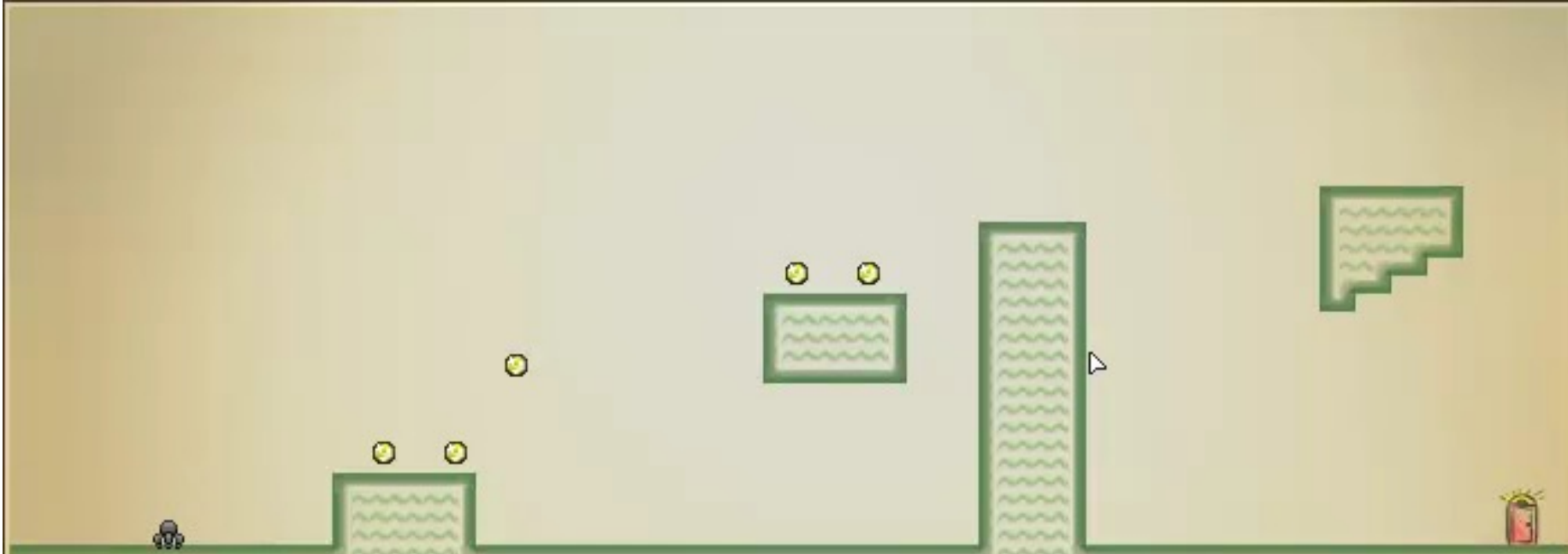


COMBO(C) REWIND(V)
QUIT RESTART(R)

TIME: 7
PAR TIME: 45
SPEED TIME: 12

ALPHA

COINS 🟡: 1/6
STARS ⭐: 0
POINTS: 0



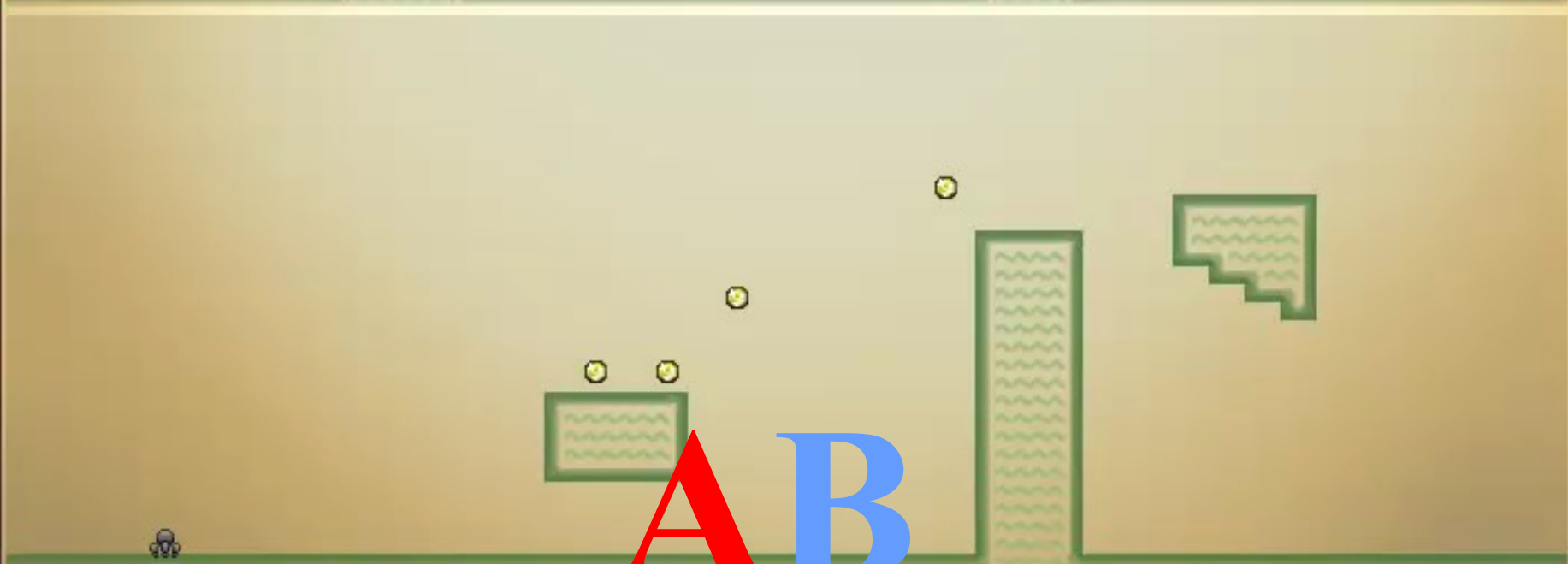
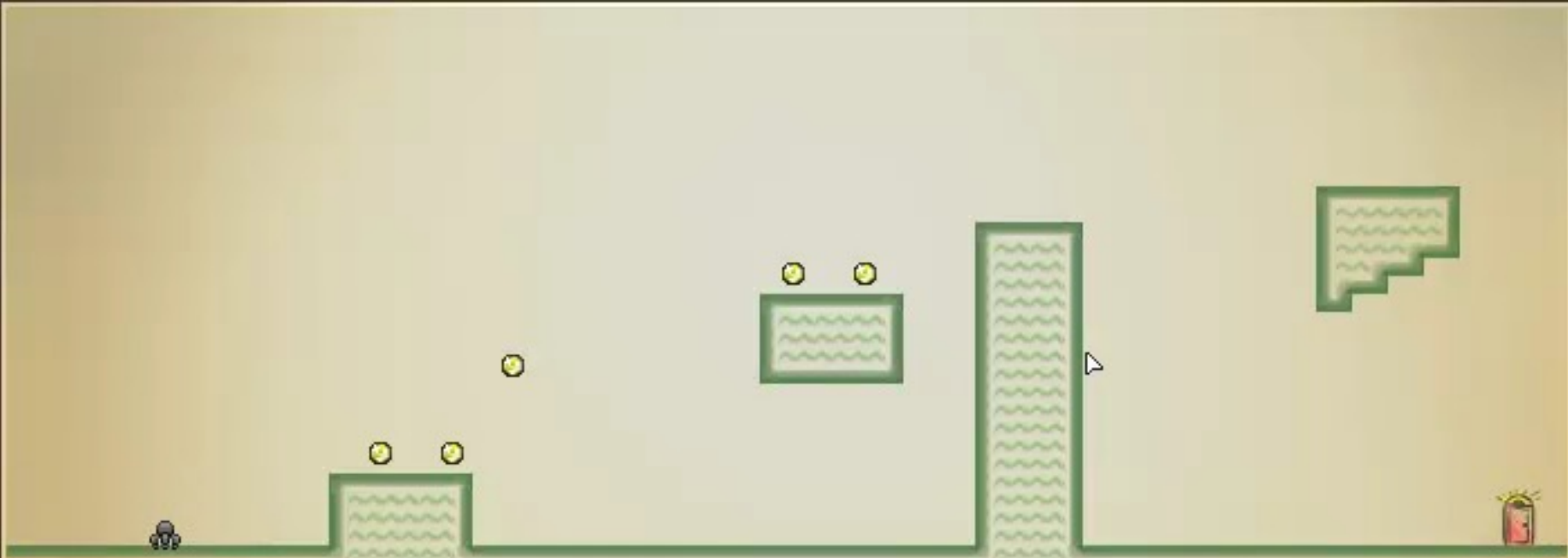
COMBO(C) REWIND(V)
QUIT RESTART(R)



TIME: 0
PAR TIME: 60
SPEED TIME: 10

BETA

COINS 0/9
STARS 3
POINTS: 255



AB

COMBO(C) REWIND(V) QUIT RESTART(R)

TIME: 0
PAR TIME: 60
SPEED TIME: 10

BETA

COINS 0/9
STARS 3
POINTS: 255



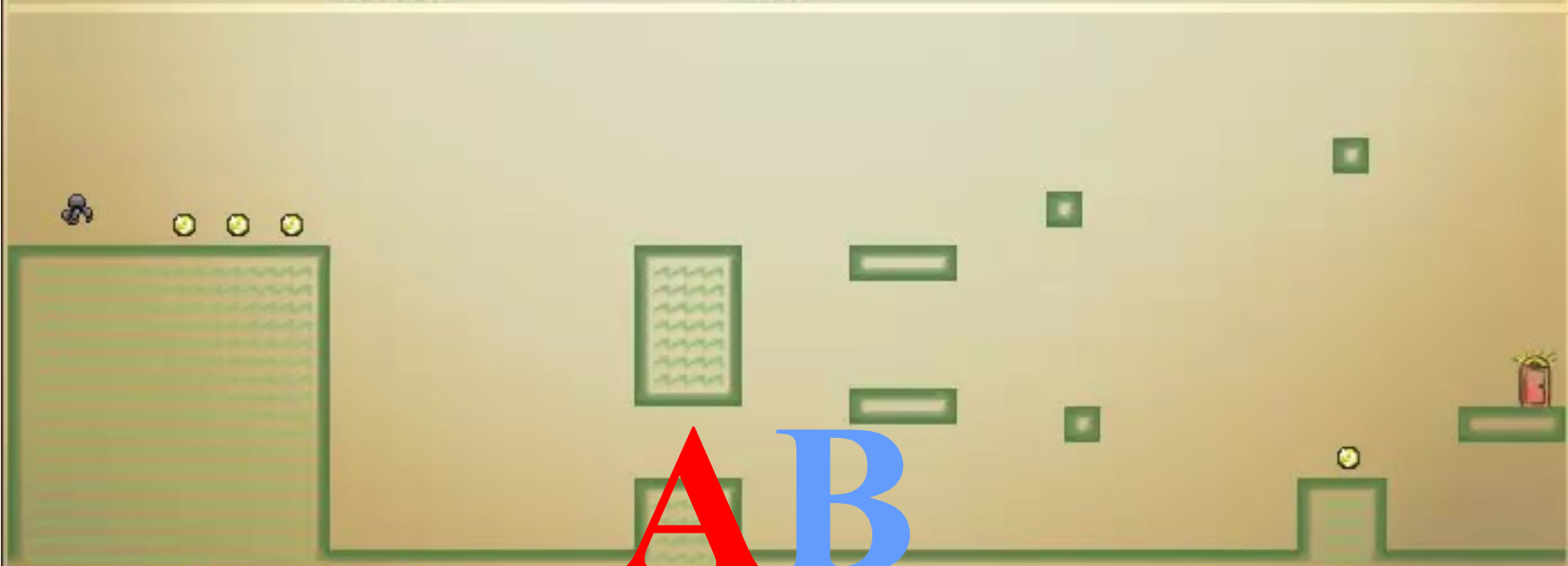
COMBO(C) REWIND(V)
QUIT RESTART(R)



TIME: 0
PAR TIME: 60
SPEED TIME: 15

GAMMA

COINS 🟡: 0/6
STARS ⭐: 6
POINTS: 573



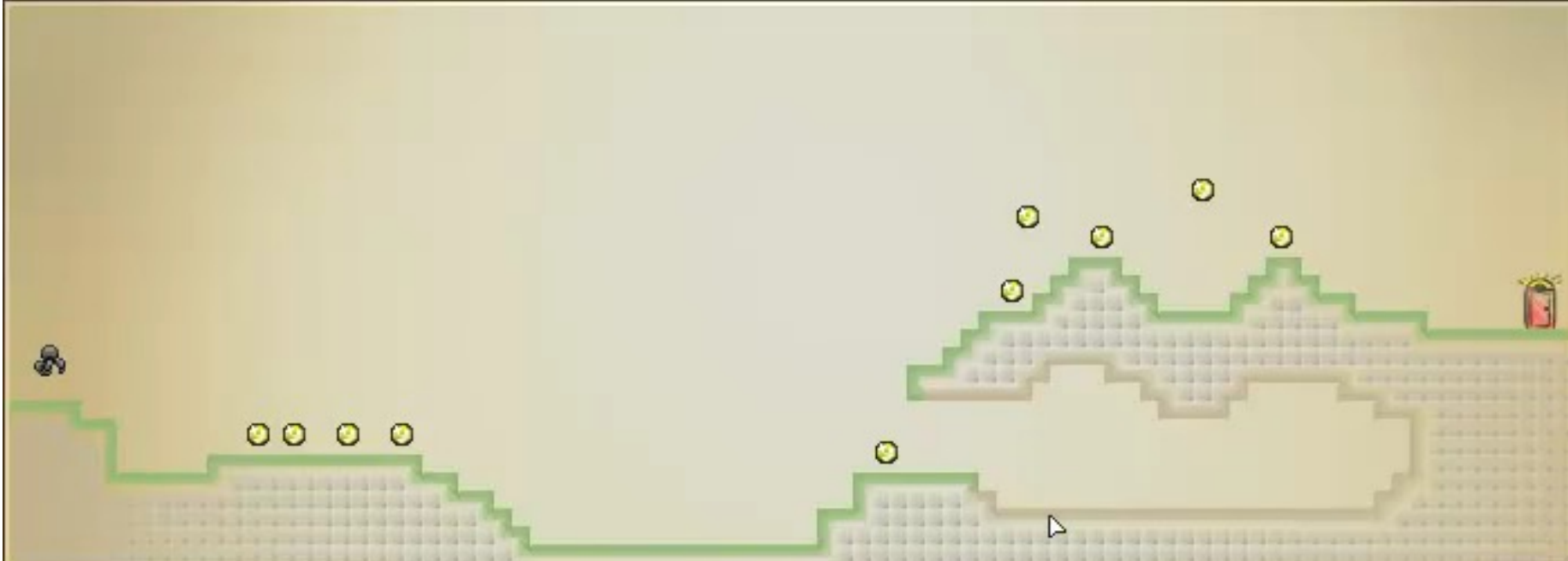
AB

COMBO(C) REWIND(V)
QUIT RESTART(R)

TIME: 0
PAR TIME: 60
SPEED TIME: 15

GAMMA

COINS 0/6
STARS 6
POINTS: 573



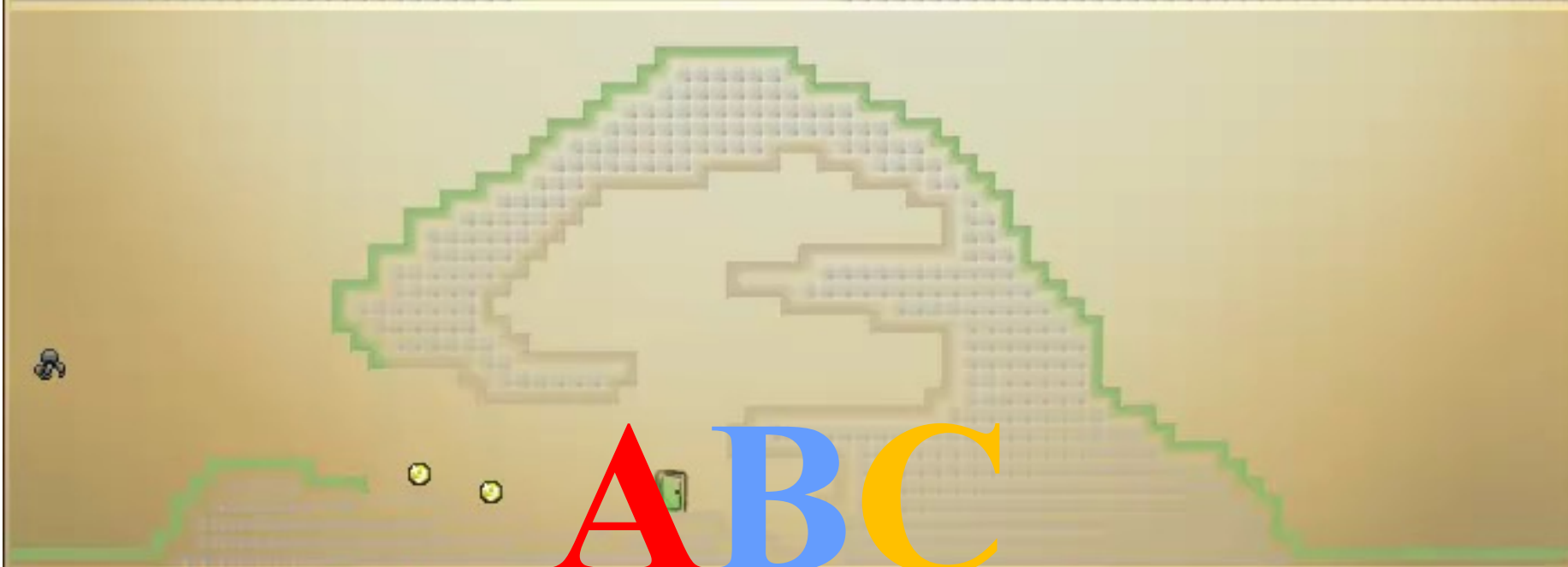
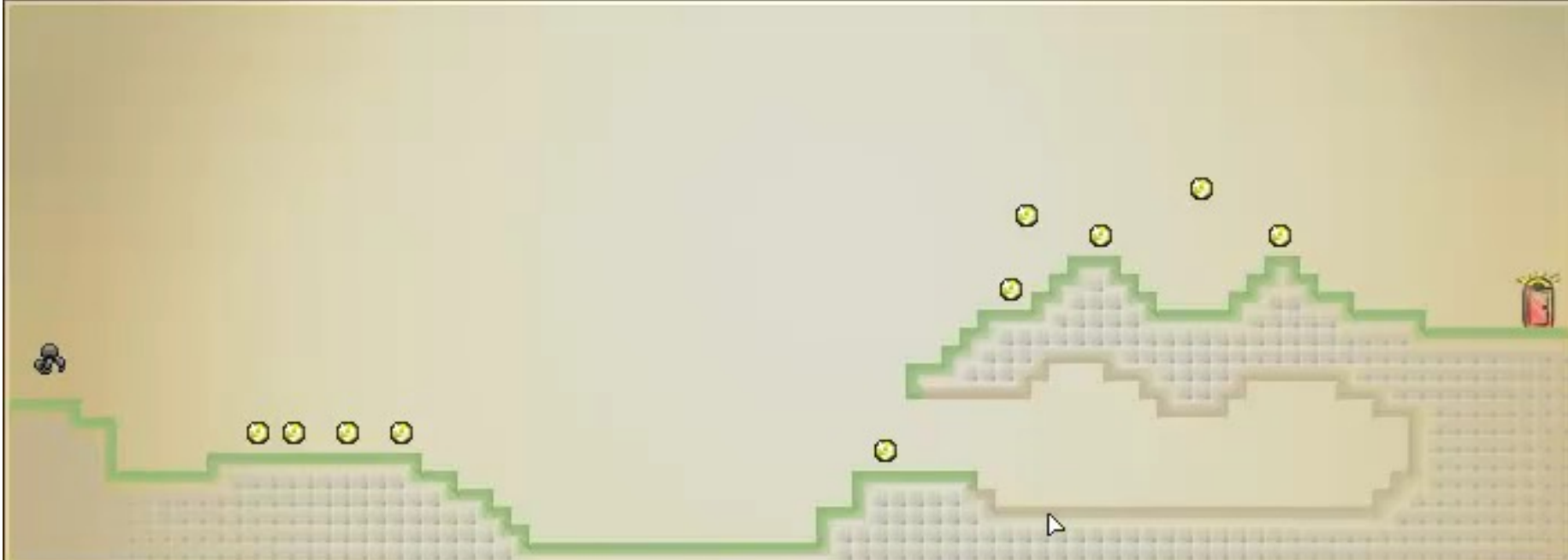
COMBO(C) REWIND(V)
QUIT RESTART(R)



TIME: 0
PAR TIME: 60
SPEED TIME: 26

MOUNTAINSIDE

COINS 0/12
STARS 9
POINTS: 879

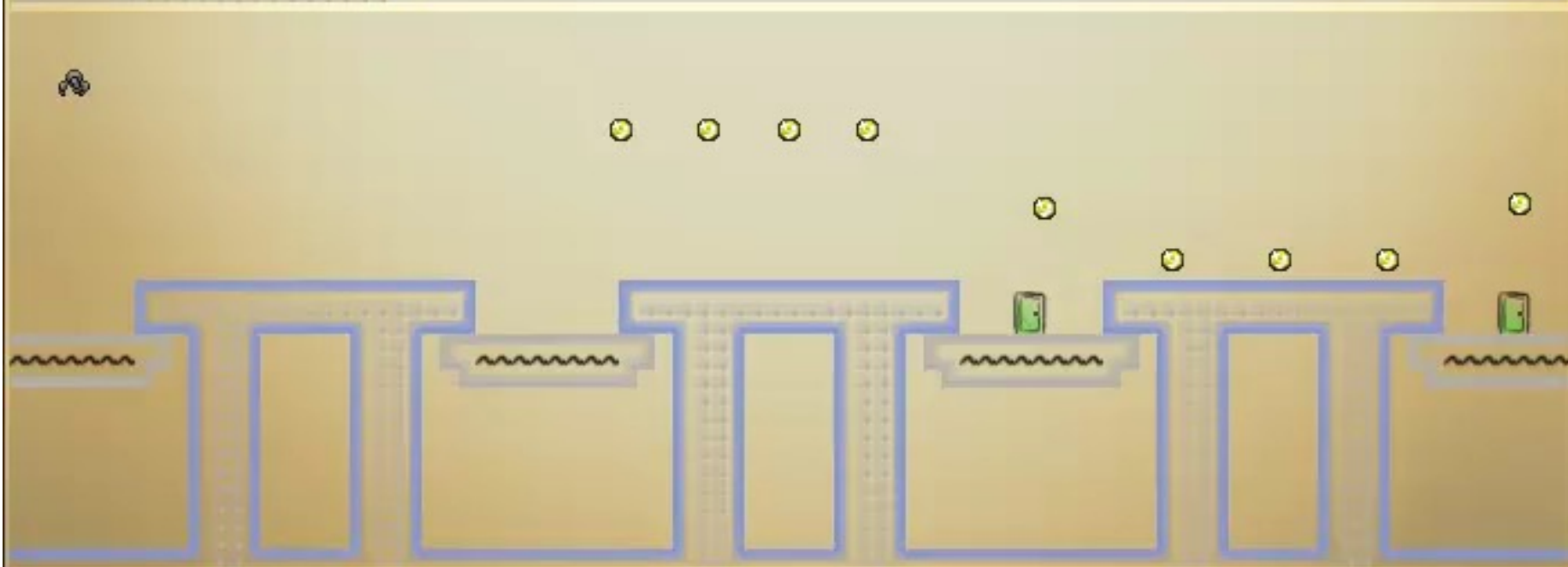
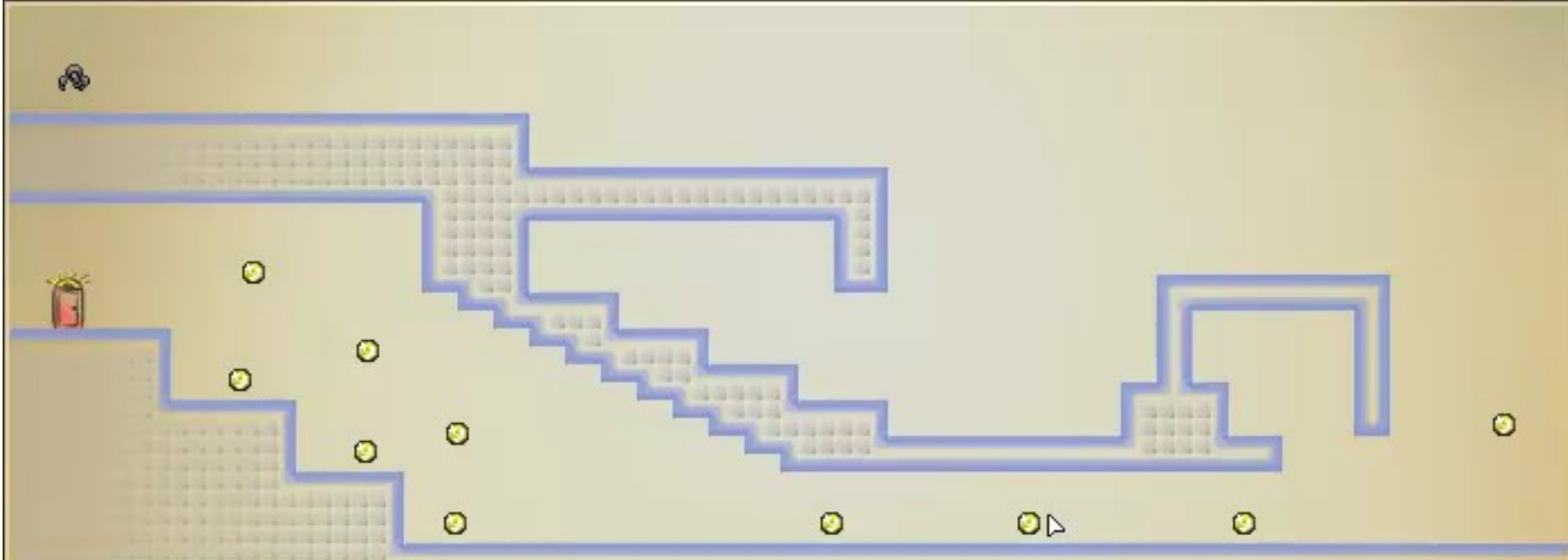


COMBO(C) REWIND(V)
QUIT RESTART(R)

TIME: 0
PAR TIME: 60
SPEED TIME: 26

MOUNTAINSIDE

COINS 0/12
STARS 9
POINTS: 879



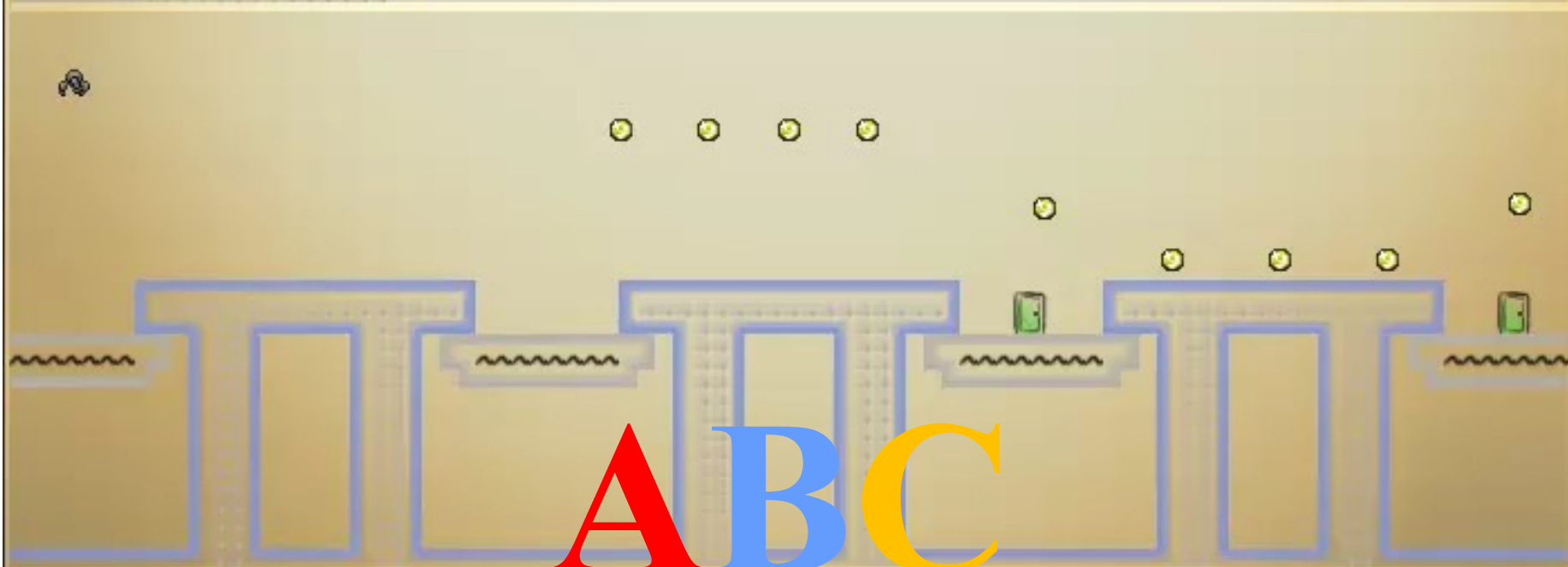
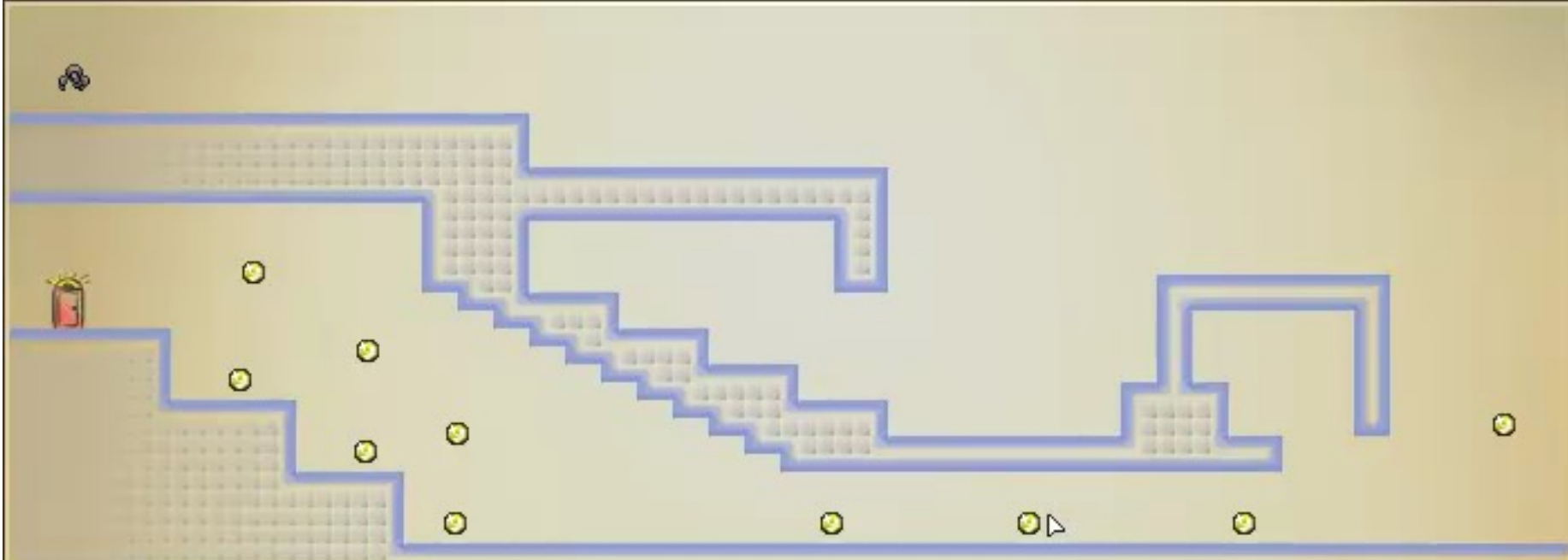
COMBO(C) REWIND(V)
QUIT RESTART(R)



TIME: 0
PAR TIME: 60
SPEED TIME: 30

PILLARS

COINS 0/19
STARS 12
POINTS: 1203



A B C

COMBO(C) REWIND(V)
QUIT RESTART(R)

TIME: 0
PAR TIME: 60
SPEED TIME: 30

PILLARS

COINS 0/19
STARS 12
POINTS: 1203

Hello Worlds

Mechanics:

A = move **B** = two worlds **C** = close world

A

AB

AB

ABC

ABC

Moderate reinforcement, high recombination

Starcraft



Starcraft

A AB ABC ABCD

Low reinforcement, high recombination

A B C D

A A A A

Summary

- Level design is always important
 - How keep your game different, lively?
 - How do you train your player?
- Level design uses **geographic constraints**
 - Create challenges by defining *design patterns*
 - Storyboard so player must go through challenges
- Level design uses **ludic constraints**
 - Do not introduce all of your capabilities at once
 - Leverage *reinforcement* and *recombination*