Adventure as a Video Game: Adventure for the Atari 2600

Warren Robinett

Context

This essay is a revised version of Chapter 3 from my unpublished book manuscript **Inventing the Adventure Game**, *which I wrote in 1983–'84. I had designed the games Adventure for the Atari 2600 videogame console (developed during 1978–'79) and Rocky's Boots for the Apple II personal computer (developed during 1980–'82).* **Inventing** *was a design history of these two commercial games. For* **The Game Design Reader**, *I added a brief introduction.*

Game Design Models

Game Spaces

Warren Robinett in 1979 designed the first action-adventure video game, Adventure, for the Atari 2600 video game console. He cofounded the Learning Company, and there designed the educational simulation Rocky's Boots. He did research in virtual reality at NASA and University of North Carolina, and now works at Hewlett-Packard Labs on fault-tolerant computing for nano-electronics.

Introduction

Game designers decide what ideas to develop based on the milieu of the time they live in—the hit games, the cool new ideas, the technical tricks for exploiting the hardware, the user interface techniques, and so on. The art of interactive game design was evolving as new ideas were discovered, and these two games of mine were links in the chain. Adventure for the Atari 2600 was directly inspired by the original text game Adventure created by Willie Crowther and Don Woods. (Chapter 2 of Inventing the Adventure Game was about this game.) Rocky's Boots (Chapter 4 of Inventing) was, for me, a logical further development of the ideas in the Atari 2600 Adventure. Many subsequent designers found the concept of the adventure game worthy of further exploration and development, producing what we now recognize as the genre of adventure games. Chapter 3 is thus a contemporaneous report, from the dawn of the video game era, on the creation of the adventure game genre. My website <WarrenRobinett.com> has more information on the design of Adventure, including the complete text of the book manuscript.

For Chapter 3 to make sense in isolation, it is necessary to know something about the game that preceded and inspired my version of Adventure. Crowther and Woods's wonderful text adventure game (see <www.rickadams.org/adventure>) used no graphics at all—it was entirely *text. Text described where you were:* `You are in a debris room filled with stuff washed in from the surface. A low wide passage with cobbles becomes plugged with mud and debris here, but an awkward canyon leads upward and west. A note on the wall says "MAGIC WORD XYZZY."` *Text described objects you could carry:* `A three-foot black rod with a rusty star on an end lies nearby.` *And text commands were typed by the player to move around and do things:* `GO WEST` *or* `TAKE ROD` *or* `SAY XYZZY`*. Chapter 3 describes how I adapted the adventure game concept from its birth medium (text descriptions of places and objects, typing text commands on a keyboard) to the video game medium (graphics, motion, animation, color, sounds, and joystick input).*

The process of creating video games has changed almost beyond recognition in the 25 years since Adventure was created. It has morphed from an act of individual authorship, similar to that of a novelist, to the coordinated effort of a large team of *specialists, similar to that of making a movie. The resources available to the game designer (memory, computing power) have also changed enormously during this period, increasing by a factor of 1000 or more. It is therefore difficult nowadays to imagine the world that we game designers lived in back then.*

The Atari 2600 was the first widely distributed video game console. After the success of Pong—the first mass-market video game—the Atari 2600 was the first home videogame machine meant to play more than one game. Different cartridges let you play different games. Today's Playstation, Xbox, and GameCube are direct descendants of the Atari 2600.

Back at Atari in the late 1970s, each game cartridge for the Atari 2600 console was created by one person. You had the idea, wrote the program, created the graphics, did the sound effects, chased down bugs, tested the game on kids, revised it until you were satisfied, and wrote a draft of the game manual. This made sense at that time, because with only 4K of ROM memory available to hold the game program, it took only a few months of programming to fill up the ROM. RAM memory was even more limited, with only 128 bytes available. (Current personal computers typically have 256 million bytes of RAM.) And not only was the memory extremely limited, but the processing power was also very limited—the Atari 2600 had an 8-bit processor with a 1.2 MHz clock speed. (Current personal computers typically have 1000 MHz clocks.) To top it off, the display hardware, although flexible, was also extremely limited, providing only two decent sprites for displaying moving objects on the screen. So you needed to start out with a game concept that was simple enough, in both graphics and gameplay, to fit into the tiny memory available. You programmed the game in assembly language. You counted bytes and machine cycles. You had to make every bit and every machine cycle carry its weight. Your job was to make the trade-offs, and come up with an interesting game, given these resources. The adventure game concept appeared to be too complex in fit into a 4K game cartridge. At least, my boss at Atari thought so. After all, Crowther and Woods's text adventure game ran on mainframe computers and required more than 100K of memory. But I thought I could do it.

Although Chapter 3 of my book is mostly about the intellectual aspect of Adventure—how I juggled the ideas and technical limits to make the game—behind the intellectual is the emotional: the motivation, the driving force. In my case, it was a combination of passion and stubbornness, taking root in the Atari culture of that period, where designer/programmers were encouraged to have their own game ideas and then code them up. The passion came from perceiving new possibilities that demanded to be explored. The stubbornness was just one of my traits. I had to fight to create Adventure. And yes, it was a good idea. It did fit into 4K. The Adventure cartridge was marketed, and one million copies were sold.

I am proud of Adventure. I'm glad I was lucky enough to be there at Atari at that time, and to have played Crowther and Woods's game. I'm glad I had the idea of an adventure game as a video game, and that I had what it took—guts, training, tools, luck in navigating the political currents—to stay with it. I hope I've steered a middle course between false modesty and arrogance. In truth, it takes a certain amount of arrogance to even try.

When you come up with your own good idea for a game, think about it for a while. Not every idea is earth-shaking. If you really feel you have a good one, don't let the Big Guys stop you. (If, on the other hand, your big idea is another DOOM clone, please go stick your head in the toilet now, and flush.) The field of interactive games is still pretty young, and I believe there are many interesting directions that are still completely unexplored. After you read my story, get busy and make something cool.

—Warren Robinett, November 2004

Description of the Game

Adventure, a video game cartridge for the Atari 2600 video game console, was the first action-adventure video game. It was published by Atari Inc. in 1978, and sold 1 million copies.

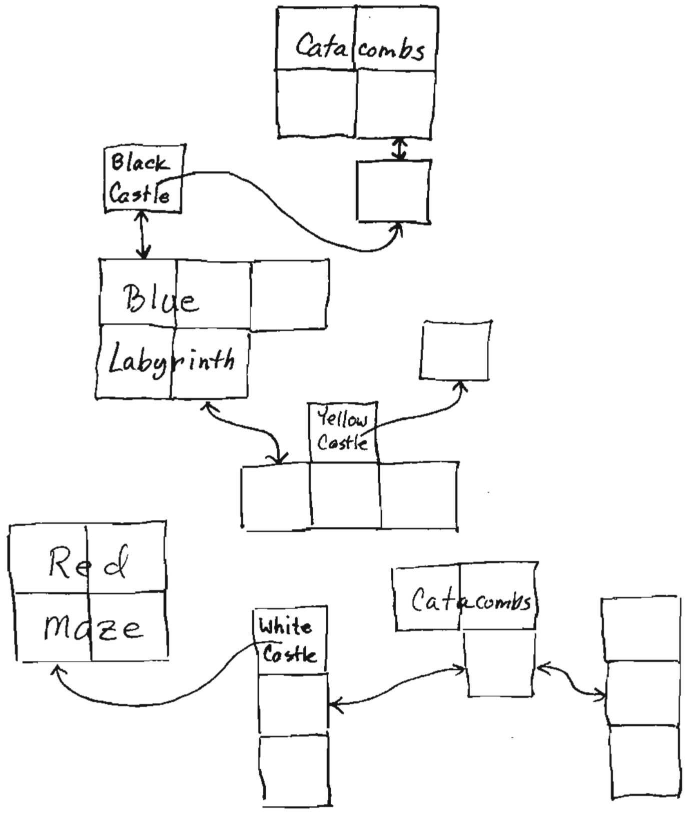
Adventure introduced the idea of movable objects (represented by visible icons) that could be picked up by the player (using a joystick) and moved from place to place in the game world. It was also one of the first video games to allow the player to explore a large multi-screen game world. As the first action-adventure game, it inspired the current genre of action-adventure video games, including *Legend of Zelda*, the *Ultima* series, and many others. Adventure also contained the first Easter Egg (hidden surprise) in a video game, which in this case was the author's signature hidden in a secret room.

I had played the new sensation, the original text game Adventure, when I finished designing my first video game. The time was June 1978, I worked for Atari, and my next order of business was to begin working on another game cartridge for the Atari 2600 home video game. I had a scheme for adapting the text dialogue of Adventure into a video game: use the joystick to move around, show one room at a time on the video screen, and show objects in the room as little shapes. I hoped the program to do this might somehow fit into the tiny (4K) memory available in a game cartridge, so, despite my boss's skepticism, my infatuation with Adventure swept me into a mad frenzy of programming.

A month later, I had a prototype: the player could move a small square "cursor" from screen to screen, picking up the little colored shapes to be found on some of the screens, which were connected edge to edge. And there was a pesky dragon that chased the cursor around, trying to eat it. Exhausted, I went on vacation, and found, on my return, that Atari upper management had decided that I should turn my fledgling adventure game into a video game about Superman. Atari's parent corporation, Warner Communication, owned the soon-to-be-released Superman movie, and a Superman video game could ride on the wave of "hype." I squirmed, and soon wriggled out of that assignment: my coworker John Dunn agreed to take over and turn the program into a Superman videogame, leaving me free to develop the same program in a different direction, namely, to continue with Adventure. (This sort of contrariness later caused Atari executives to label their video game designers "a bunch of high-strung prima donnas.") I moved forward, discovering how ideas from the text adventure game could be made to work as moving shapes in a video game, and discovering what the young tradition of video games could contribute to the new genre of adventure games. The program took eight months to complete start to finish; Atari marketed the cartridge, and since Woods' game was in the public domain, the video game, too, could be called Adventure.

This new videogame version of Adventure was a quest: the player started out beside the Yellow Castle with the goal of retrieving the Enchanted Chalice, which was out there somewhere in the network of thirty rooms. (See *Figure 1*). To make things difficult, three dragons infested the game, chasing the player from room to room, and trying to eat him. A giant bat also caused trouble, moving objects around and stealing things from the player. There were a number of useful objects. The sword killed dragons. The bridge let the player cross walls in the maze. The magnet sucked out objects that were stuck in the walls. The black, white, and yellow keys each unlocked a castle of matching color. (See *Figure 2*).

Figure 1: Map of the Network of Rooms



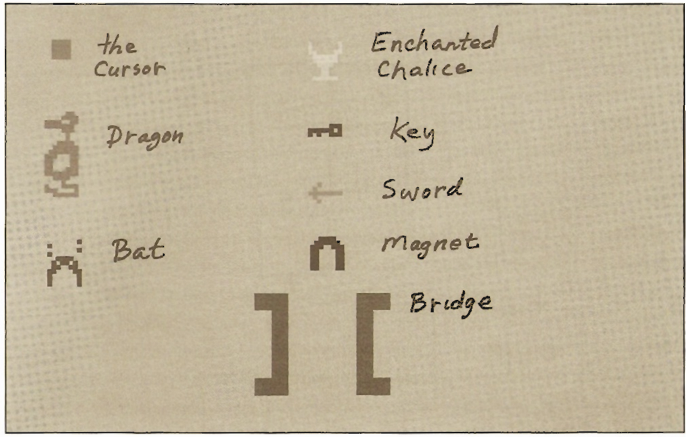


Figure 2: Creatures and Objects

Each of the three castles in Adventure had interiors composed of one or more rooms. The player entered a castle by moving through a doorway equipped with a portcullis, which could be raised or lowered with a key. When the portcullis was down, the castle was locked, and entering or leaving the locked castle was impossible. Each of the three castles had its own key object, and all the castles were locked when the game began.

This video game, Atari 2600 Adventure, was inspired directly by Crowther and Woods' text game, Adventure. I tried at first to create video game counterparts of features in the text game. The magic rod can create a crystal bridge to span an impassable crevasse in the text version; I tried a rod shape which that when it touched a maze wall, caused a bridge shape to appear. The "maze of twisty little passages, all alike" became a very confusing 8-room video maze. These direct transliterations from text to video format didn't work out very well. While the general idea of a video game with rooms and objects seemed to be a good one, the graphic language of the video game and the verbal language of the text dialogue turned out to have significantly different strengths. Just as differences between filmed and live performance caused the art form of cinema to slowly diverge from its parent, drama, differences between the medium of animated graphics and the medium of text have caused the animated adventure game to diverge from the text adventure game.

Player Input

The adaptation of the adventure game to the video game medium required a radical change in the form of the player's commands to the game. Typing was not possible on the Atari 2600 video game console: the standard input was a joystick with one "fire" button on it. The video game player could push the joystick lever in one of four directions, or press the button. The text game player, on the other hand, typed in a two-word command, composed of an action verb and a noun. There were dozens of words in the text game's vocabulary, both for nouns and for verbs; in two-word combinations, there were thousands of possible commands. How could the video game player initiate the wide variety of actions that were possible in the text game?

The joystick was a natural for north-south-east-west movement. There is something satisfyingly responsive about shoving the joystick lever and having a shape on the screen move in the same direction. It is important that players can hold onto the single lever and move it in any of several different directions. The lever itself fades out of consciousness, and players feel that they are propelling the cursor with their own muscles, as if they were scooting a brick around on a sidewalk.

Indeed, players identify themselves with the shape they move around on the video screen. When they say, "I ran into a wall," they mean the shape they moved ran into a wall; they *are* that shape. In Atari 2600 Adventure, this self shape is a little solid-colored square. It can be called a "cursor," since its function, as a position indicator is similar to the rectangular blinking cursor found on word processing screens. I originally called it "the man."

Besides movement, picking up and dropping objects are the most important player actions in an adventure game. With the joystick lever assigned to movement, the single button on the base of the Atari joystick was the clear candidate for grappling with objects, although it wasn't clear exactly how the button should control taking and dropping objects. If the function of "take," for instance, was to be invoked when the button was pressed, which object should be taken? The graphical nature of the video game provided a solution to this. In the world of video games, objects on the screen usually interact only when they run into each other. This is called a "collision," and is defined as an overlap of one shape with another. The object shape to be taken could be specified by moving the cursor to touch it. In fact, the collision itself could invoke the pick-it-up action, which left the button free for dropping. But dropping what? If several objects had been picked up, a selection was again needed. And how should the carried objects be shown? Although pushing the button could have called up an "inventory" screen, and the cursor could have been steered into the object to be dropped, a simpler solution was adopted. Only one object could be carried at a time, and that object was shown besides the cursor on the screen. Pushing the button dropped it.

This approach had several advantages. It was simpler, so the program was shorter. The screen always showed the room players were in, and what they carried; players didn't have to worry about their cursor being eaten by a dragon that came by while they were examining inventory. Of course, time could have been suspended during inventory view, but that didn't seem right for a real-time game. The limitation of being able to carry only one object gave players some interesting strategic choices: which object should they carry—the treasure or the weapon?

Since the object being carried was shown on the screen, it had a position relative to the cursor. Players could adjust the positioning of their held object. When exploring unknown dragon-infested territory, it usually made sense to have your sword out front, because simply holding a sword did not prevent a dragon from eating you—but poking the sword into the dragon did. For a dragon that was scared of the sword, it worked better to loiter near a room boundary, with a behind-the-back sword dangling into the next room, ready to make a swift stroke when the dragon came into striking range.

What about all the other actions that players might want to initiate in an adventure game? These, too, could be specified by touching objects together. For example, in a text game, one might command "KILL DRAGON." The corresponding action in the video game was to pick up a sword and touch it to the dragon. In a sense, the held object and the touched object were the analogues of the action verb and noun from the text adventure game.

A great variety of "commands" might be given if players had the right verb objects at hand. Placing the bridge object across a maze wall and going across it was equivalent to "CROSS WALL." Touching a key to a castle's portcullis commanded, "UNLOCK CASTLE." Bringing the magnet into a room to retrieve a sword stuck in the wall was like "ATTRACT SWORD." Thus the syntax of nouns and verbs in the text adventure had an analogue in a video adventure—a "syntax" of overlapping shapes.

Objects

Objects that could be picked up, carried from place to place, and used for various tasks, were described with a phrase in the original text game Adventure:

```
A three-foot black rod with a rusty star on the end lies nearby.
```

Such an object became, in the graphic language of the video adventure game, a little colored shape that appeared at some location on the screen. Each object had a location, which consisted of a room and an (X, Y) position within that room.

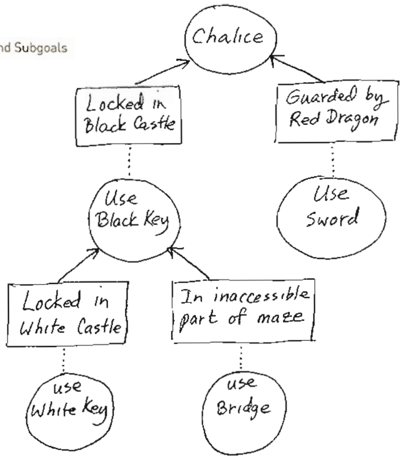
Each object in Atari 2600 Adventure does something. The keys open castles; the sword kills dragons; the bridge crosses walls; the magnet retrieves lost objects; and the chalice wins the game. Although it would of course be possible to have useless objects in a game, to serve as decoys or decorations, in Adventure every object has a function. The objects are really tools since players can use them to cause things to happen in the game world.

Whereas the goal of the original text game Adventure was treasure gathering, the video game Adventure is defined as a quest. One single treasure, the Enchanted Chalice, must be located and brought home. Thus, the tool objects must contribute somehow to the overall goal of the quest. For example, if the chalice is locked inside the Black Castle, then finding the Black Key becomes a subgoal, subordinated to the primary goal of getting to the chalice. If the Black Key is found, but is inaccessible because of the dragon guarding it, then another subgoal is spawned—find the sword so as to get past the dragon. Each tool object is a means of getting past a certain kind of barrier. Since needed objects may be behind barriers that, in turn, require other objects, a hierarchy is created of goals and subgoals. *Figure 3* shows the arrangements of some obstacles and their solutions.

Creatures

A creature in an adventure game is an object that moves around on its own, initiating actions. It is best to consider a creature as a special type of object so that the creature can inherit the traits already defined for objects: shape, color, location, and ability to be picked up. Each type of creature has some special rules that specify how it behaves, what it responds to, and what actions it can take. These special rules are defined by a part of the game program, usually a subroutine that corresponds to the creature type. There are two species of creature in Adventure: dragon and bat.

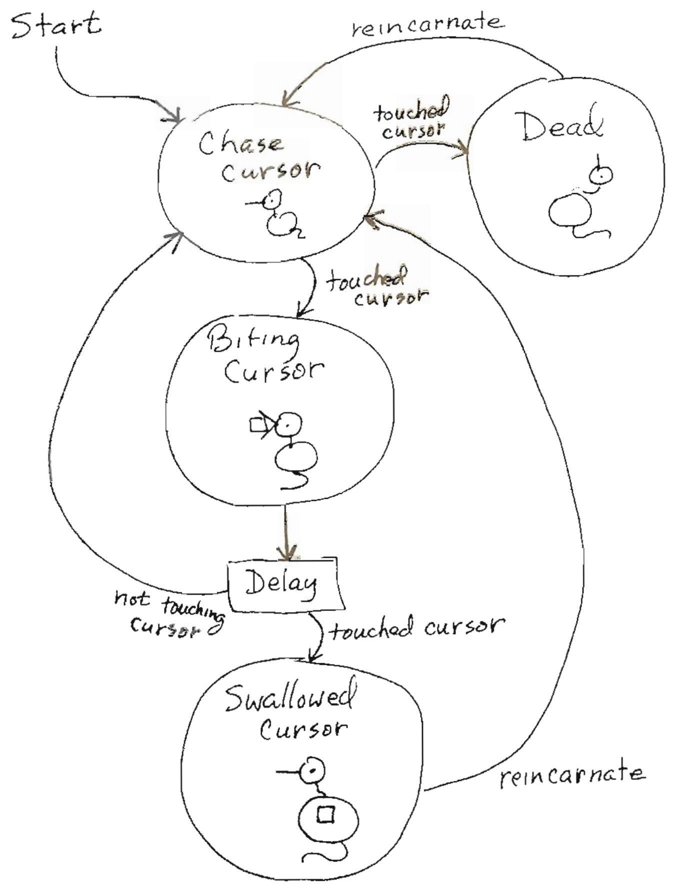
Figure 3: Hierarchy of Goals and Subgoals



Dragons are the main villains in the game. There are three of them, and they chase the player's cursor around, trying to eat it. If players' reflexes are too slow, or if they get cornered, the dragon swallows their cursor. Once eaten, a player's cursor can be "reincarnated" to get out of the dragon's belly, but as a penalty is sent back to the starting location, losing whatever it was carrying, and any dragons it may have killed are reincarnated, too. (Having already killed some dragons is like being vulnerable in the game of bridge. There is more to lose in the battle with the third dragon. Getting eaten means that the two dead dragons will come back to life.)

There are four states that a dragon can be in, each shown on the screen by a different dragon image: chasing a player's cursor, biting it, having swallowed it, and being dead. The state diagram of *Figure 4* shows the conditions in the game that cause transition from one dragon state to another. In a typical interaction between player's cursor and dragon, the dragon goes back and forth between the biting and chase states several times as it tries to eat the player's cursor, and then the cursor either gets away, gets eaten, or kills the dragon with the sword. The dragon graphics change rapidly, mirroring the state changes. Not only is this interesting animation, but it also gives players valuable visual feedback about which state the dragon is in at any given moment.

Figure 4: State Diagram of Dragon



For the dragon to succeed in swallowing a player's cursor, it must collide once with the cursor (entering biting state), and after a fraction of a second, collide again. This brief delay gives players time to recoil, and if their reflexes are fast enough, then they avoid having their cursor swallowed and the dragon resumes chasing it. As a player's cursor and dragon go repeatedly through this chase-bite cycle, neither has the advantage. The dragon wins the battle if it can swallow the player's cursor; the player wins if he or she can get to a sword and use it to kill the dragon.

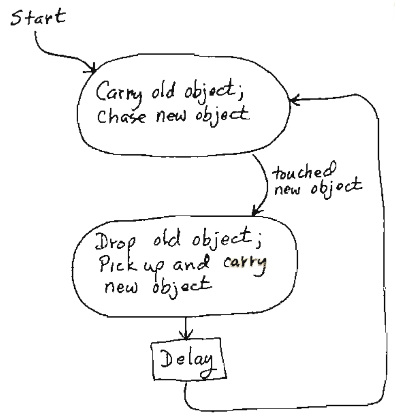
It is significant that it takes two collisions with the dragon, not one, for the cursor to be swallowed. Merely colliding with the dragon is not fatal. Thus the interesting chase-bite cycle is made possible. Most video games define simple one-time collision with enemies as fatal and irrevocable, and thereby miss a chance to create a more interesting interaction.

The length of the recoil interval between biting and swallowing is quite important. If it is too long, it is trivial to avoid being eaten, and players can ignore the dragon and do whatever they want. If the interval is too short, players never succeed in recoiling, and their cursors get eaten every time. There is a middle ground between "trivial" and "impossible" called "challenging." Trying out the game with various players and watching how well they do is the best way to adjust a game's timing. This process is called "tuning" the game. Varying the length of the recoil interval turned out to be an effective means of varying the game's difficulty. It ranges from around a tenth of a second at the most difficult, to about three seconds at the easiest.

The bat is the second species of creature in Adventure. As it flaps from room to room carrying along an object, periodically the bat tires of its current trinket, and discards it in favor of a new object to carry off. Without the bat, non-creature objects would never move from the spots where the player dropped them. The effect of the bat is to move objects around, to disturb the predictability of the game. The bat is the game's confusion factor.

As detailed in *Figure 5*, the bat has two states: seeking a new object to pick up, and carrying off a newly acquired object, ignoring all other objects. This ignore state, which lasts for about ten seconds after a new object is picked up, was needed to make sure the bat would carry its new trinket off to another room. In an early version of the game, the bat reentered the seek state immediately after picking up a new object; in a room containing two objects (plus the one carried by the bat) the bat would ferry objects back and forth between the two positions forever, never leaving the room.

e 5: State Diagram of Bat


Start
Carry old object; Chase new object
touched new object
Drop old object; Pick up and carry new object
Delay

The bat can carry dragons. It sometimes happens that the bat will appear carrying a dragon, steal the player's sword and fly off with it, leaving the disarmed player to deal with the left-behind dragon.

Mazes

In a text adventure game, a room is a single location. Although there are passages to other rooms, the room itself has no internal structure. A video adventure game, by comparison, allows the player to have a position within a room, shown on the screen by the cursor's position. A single room can show a simple maze on the screen, with passages going off the screen to other (as yet unseen) maze rooms. The walls of the maze, of course, block the cursor's movement. A 4- or 5-room maze can be quite complicated.

Besides forming mazes, walls prevent the player from leaving a room in certain directions, and thus help form the overall layout of the network of rooms. In some rooms in Crowther and Woods's text adventure game, typing "GO EAST" would produce this response.

There is no way to go that direction.

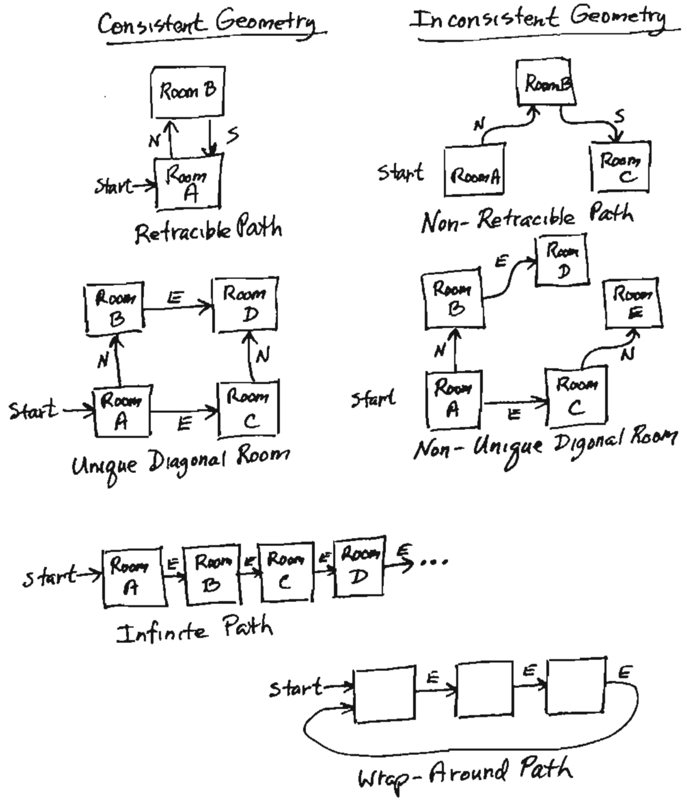
Walls perform the analogous function in a graphical adventure game.

A maze is a geometric construction in space; the positioning of its walls defines a maze. Video graphics do an excellent job of capturing the geometry of a maze. By contrast, using sentences to describe a maze is inefficient and piecemeal. Players lost in the maze of "twisty little passages" in Crowther and Woods's text adventure game invariably draw a map if they want to get out by understanding the maze, rather than by just the luck of random wandering. The verbal representation of a maze is abstract, whereas the graphical representation is concrete and therefore less confusing. Five-year-olds learn the path to the Black Castle through the five maze rooms of the Blue Labyrinth in Adventure. Moving through the visual depiction of each room lets children remember their path in the same way that they remember the route from living room to bathroom in a friend's house.

One leaves a room in video Adventure by driving off the edge of the screen. Since the screen has four edges (top, bottom, left, and right), every room has four links to other rooms (to the north, south, east, and west). Going from one room to another in an adventure game is like using a Star Trek transporter. ("Beam me up, Scottie!") The player vanishes from one place and materializes in another place. The two places are not necessarily "near" each other: they are merely connected. It is often impossible to draw a map of an adventure game's network of rooms so that all linked rooms are side by side. The map of Adventure (*Figure 1*) is an example of that: although small groups of rooms can show links among each other by adjacency, there are leftover links that must be shown explicitly with lines running between the two connected places. If a map, which is a shrunken replica of a region's geometry, cannot be constructed, then that region cannot be built full-sized, either. In other words, these places are impossible. Adventure games simulate spaces that can't exist in the physical world. But it doesn't really matter that a map or scale model of these places cannot be built; players can still move from room to room in the game. As in the Bugs Bunny cartoon with the vast volume inside the tiny tent, these impossible spaces have surprising properties.

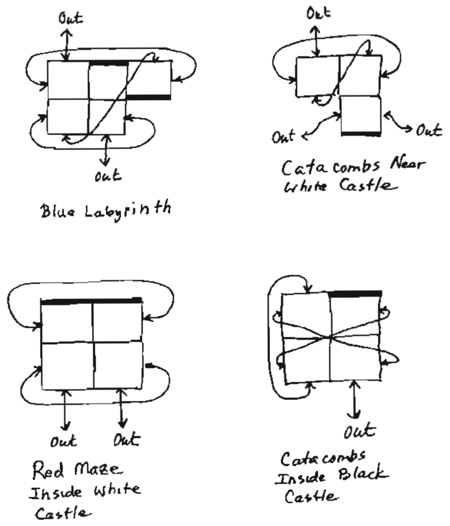
A network of rooms, depending on how the rooms are interconnected, can have inconsistent geometry. One might assume that a lot of square screens connected edge to edge could be thought of as a big array of screens, arranged as rows and columns in a plane, just like the square tiles on a kitchen's linoleum floor. However, rooms can be interconnected in a way that is inconsistent with plane geometry, and with commonsense expectations about moving through space. Three inconsistencies that occur are non-retraceable paths, nonunique diagonal rooms, and wrap-around paths. (See *Figure 6*).

Figure 6: Consistent versus Inconsistent Geometry



In moving through space, common sense suggests that the space passed through is still there, and that, if necessary, one should be able to turn around and retrace one's path back through that space. Of course, this is not always true in real life: there are one-way doors that swing shut and lock, and trapdoors that one can fall through and not be able to climb out of. If players can go one room north, then go south from that room and be back where they started, they have retraced their path. If the room-to-room south link doesn't take them back to where they started, there is no backing up—they are on a one-way, non-retraceable path. This non-retracibility is a device the designer may use to construct a trickier, more confusing maze. Because it is essential that a game be challenging, confusing players to a certain degree is a quite proper objective for the game designer. Too much illogical trickiness, however, can frustrate and irritate players. In the video game Adventure, I chose to let players always be able to retrace their path.

Figure 7: Comparison of the Room Topologies of the Four Mazes


Out
Out
Blue Labyrinth
Out
Out
Out
Catacombs Near White Castle
Out
Out
Red Maze Inside White Castle
Out
Catacombs Inside Black Castle

Does going one room north, then one room east get to the same place as east then north? The answer is either yes or no, depending on how the rooms are linked. Although on a perfectly flat plane, going east forever will never bring one back to his starting point, the surface of the earth, which seems pretty flat, wraps around to meet itself eventually . So, too, in a finite network of rooms, the limited memory of a computer would have difficulty representing an infinite number of rooms, any path must eventually return [wrap around] to some previously encountered room, assuming walls never block the path.

The pattern of interconnection within a network of rooms is called "topology." Topology has to do with what is connected to what. The four mazes of Adventure are all unrealizable in the flat. They wrap around in various strange ways [See *Figure 7*], and thus have interesting topologies.

The Blue Labyrinth

The Blue Labyrinth in Adventure has two features that make it different from mazes printed on paper: inconsistent geometry and partial view. *Figure 8* shows how the five rooms are interconnected to form the Blue Labyrinth. The maze is more confusing than the diagrams suggest because only a part of the maze [one room] can be seen at any one time. Viewing the diagram, the eye can rapidly follow maze paths and identify dead ends; but in playing the game, the cursor must move along each path to explore it, and laboriously retrace from dead ends. Not only is exploring slower, but explorers must rely on their memory of the maze rooms they have passed through in order to form a mental model of the whole maze. Players find the Blue Labyrinth quite confusing at first—more so than would be expected from the number of its forks and different passages—and the principal source of this confusion is that players get only a partial view of the maze, never seeing the whole. One is reminded of the story of four blind men examining an elephant, forming different conclusions about what kind of beast it was from feeling its trunk, tusk, foot, and tail. It is hard to reconcile several partial views of something into a coherent global picture.

To add to the confusion, unless normal assumptions about spaces are abandoned, no coherent model of the Blue Labyrinth exists. This is because the Blue Labyrinth has several wraparound paths and nonunique diagonal rooms. One player remarked that he could learn paths through the maze from place to place, but could never get a picture of the whole thing in his mind.

Figure 8: Room Topology of the Blue Labyrinth

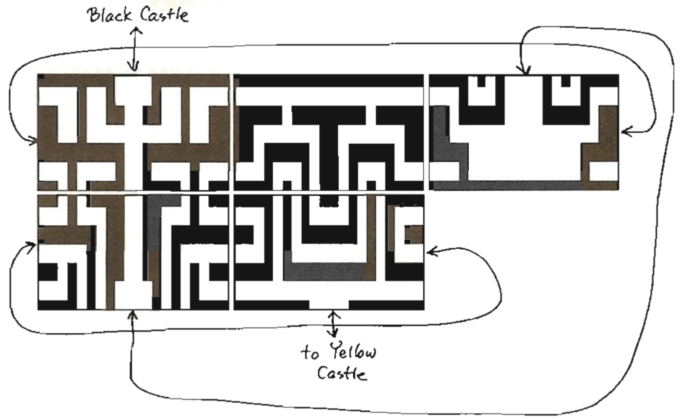
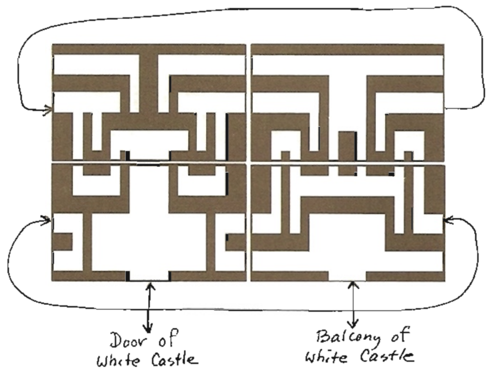


Figure 9: Room Topology of the Red Maze Inside the White Castle



These inconsistent maze geometries are confusing because players' experience with mazes (usually printed on flat pages) leads them to expect more flat mazes. They attempt to make a mental model that incorporates the maze rooms they have seen into a flat map. Even though the players explore, contradictions occur in their mental flat map; the assumption that maps are flat surfaces is a deep-seated one and hard to challenge. Experience exploring real-world mazes when surfaces are not flat at least offers a clue. But the Blue Labyrinth offers no such clue.

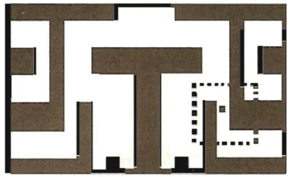
The Red Maze

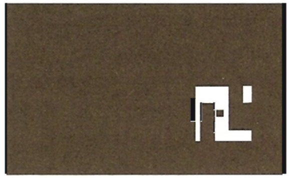
Inside the White Castle lies the Red Maze. (See *Figure 9*). The distinguishing feature of this maze is that it is composed of two disjointed sets of passages that are intertwined but do not connect. Players enter Section 1 of the maze through the door to the castle; to get into Section 2, they must bring the bridge object into the castle and use it to cross one of the maze-walls that separate the two sections.

Section 1 extends through only three of the four rooms of the Red Maze; without using the bridge, players cannot get a glimpse of the fourth room and whatever useful objects it might contain. In Game 2 of Adventure, the key to the Black Castle starts out in this hard-to-get-to place.

The topology of the Red Maze is simpler than that of the Blue Labyrinth: wrapping around in the horizontal dimension, it goes down from either of the two lower rooms to exit from the castle. One of these exits is the normal one through the door of the castle, but the other exit, from the hard-to-get-to room in Section 2, leads to the "balcony" of the White Castle. The link from the Red Maze to the balcony is one-way, however; it is not possible to go back into the Red Maze from the balcony. This violates my principle of making all paths retraceable, and was a mistake—an earlier idea that was rejected but never expunged. (Once a product is released, harmless bugs like this one are often redefined by the marketing people as "features." The manual for Adventure explains away its bugs under the heading "Bad Magic.")

Figure 10: One Maze-Room in the Catacombs


(10a) diagram of invisible part of maze


(10b) actual screen image

Catacombs

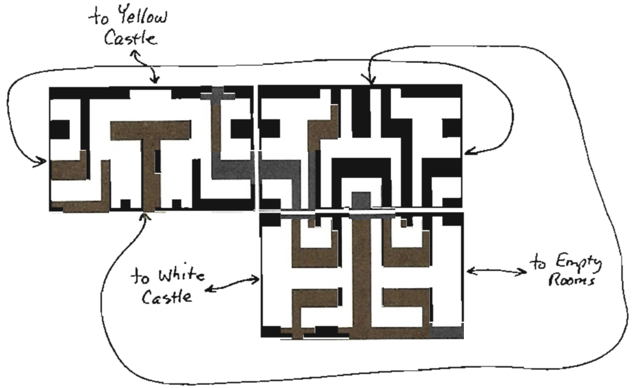
In the original text game Adventure, once players had moved a couple of rooms into the cave, they got this message:

```
It is now pitch dark. If you proceed you will likely fall into
a pit.
```

The solution was to light the "shiny brass lamp" picked up earlier. Atari 2600 Adventure has a maze that works in analogous fashion: orange lamp glow penetrates a short distance into the surrounding gloom to expose nearby maze walls. Mazes of this type are called catacombs. *Figure 10a* diagrams the walls of a typical maze room in the catacombs, and the dotted line around the cursor shows how far the lamp glow reaches. This illuminated area around the cursor is equivalent to the circle of radiance thrown out by a lantern, but a smooth circle being impossible, in this case, the circle of radiance is square. Beyond the illuminated area near the cursor, walls cannot be distinguished from passages. *Figure 10b* shows the screen image resulting from the maze position diagrammed above it. The static image does a poor job of conveying the feel of being in the catacombs; the cursor can move about the screen, bumping into unsuspected walls, and the orange firelight surrounds the cursor wherever it goes. It is like shining a flashlight around in a cave.

Just as seeing a single room of the Blue Labyrinth is a partial view of its 5-room entirety, the image of the maze walls near the cursor is a partial view of a single catacomb room. The lamp glow covers about a tenth of the screen. The two catacomb mazes in the game consist of three rooms and four rooms. So, in the catacombs, a partially viewed room, if the player could imagine it entire, is itself only a partial view of a multiroom maze. The smaller

Figure 11: Room Topology of the Catacombs, near the White Castle

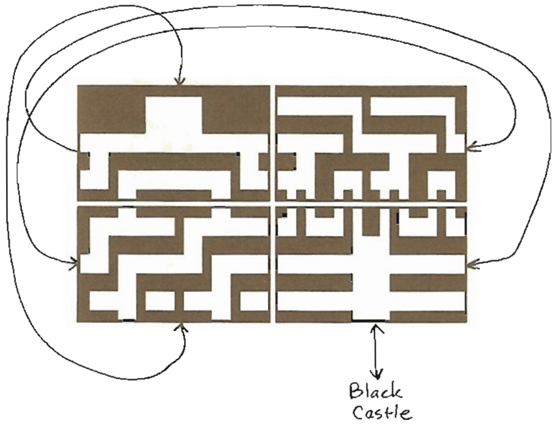


the individual views are relative to the whole, the greater the difficulty in assembling them into an overall picture. The Blue Labyrinth, with its five rooms viewed one at a time, has a fractional view ratio of 1:5. By comparison, the 3-room catacombs near the White Castle, with each room seen a tenth at a time, compounds fractional views of 1:3 and 1:10 to yield an overall ratio of 1:30. Thirty different images must be remembered (or sketched) and joined together correctly to get a picture of the entire maze. The interconnections among the three rooms of this maze are shown in *Figure 11*.

The Catacombs Inside the Black Castle

The maze inside the Black Castle is a tricky one. It is a catacomb-type maze, with the lamp glow surrounding the cursor allowing only a partial view of each room. Four rooms compose the maze, yielding a view ratio of 1:40. In addition, the topology of the maze is complicated. (See *Figure 12*.) Like the Red Maze, the Black Castle maze has two disjointed parts: the bridge is needed to get from one section to the other. But unlike the Red Maze, where the two sections were of similar size, in this maze the isolated section is a tiny little chamber at the bottom of one room. The existence of this chamber is not obvious because when the player's cursor is beside the chamber, but not in it, the orange lamp glow does not extend far enough for the

Figure 12: Room Topology of the Catacombs, inside the Black Castle



player to see that the chamber is surrounded by maze walls on four sides. The chamber (partially viewed) seems to be just the dead end of one of the other passages. And yes, there is an interesting object—the Gray Dot—hidden within this chamber.

The dot was not needed in the normal play of the game. Being gray, it was invisible until it was picked up. It was not mentioned in the game's manual because Steve Harding, who wrote the manual, didn't know it existed. The dot's purpose, which had to be discovered by trial and error, was to allow passage through a certain sidewall into a secret room. In this deviously hidden secret room, I affixed my signature, in flashing lights: "Created by Warren Robinett."

I did this in the tradition of artists, down through the centuries, identifying themselves as the authors of their own works. Atari imposed an irksome anonymity upon its designers, so subterfuge was required to put one's mark upon a game. I kept the secret of the signature room to myself for a year (not an easy thing to do) in order to avoid being forced to expunge it from the program, and also to test experimentally whether players could, on their own, discover such an obscure thing. I really wasn't sure that it would be discovered, but thought that if several hundred thousand cartridges were manufactured, then someone, somewhere, would find the dot and the secret room. I remembered, from when I was in high school, the rumors that Paul McCartney was dead, and how people played Beatle records backward, searching for secret messages.

The Adventure cartridge was manufactured and marketed, and the secret room was discovered. A fifteen-year-old from Salt Lake City wrote to Atari, explaining with a detailed diagram how to get into the secret room. By then, I no longer worked for Atari, so they had a couple of their designers track down the part of the game program that produced the secret room. Brad Stewart, who located the offending code, said that if he were assigned to change the program, he would replace "Created by Warren Robinett" with "Fixed by Brad Stewart." Ultimately, Atari blessed the whole idea, referring to hidden surprises in their games as "Easter eggs."

The major innovation of Atari 2600 Adventure is the idea of moving a cursor through a network of screens connected edge to edge. This idea made it possible to make a video game that was at the same time an adventure game, by identifying the network of screens with the adventure game's network of rooms. The action of the game could therefore take place in a much larger and more interesting space than the single screen of most of the then-current video games. It was natural to adopt the small movable shapes provided for in the video game hardware to be adventure game objects. Crowther and Woods had established in their text adventure game that objects were tools for getting past obstacles. My idea for a sequel to Adventure was to allow tools to be combined in order to solve more complex problems. I thought of this as building machines. The idea evolved as I worked on it. The end result, three years later, was a game in which tool objects (sensors, logic gates, and others) could be combined to make machines that made things happen in the game in response to conditions detected by the sensors. This game was called Rocky's Boots.