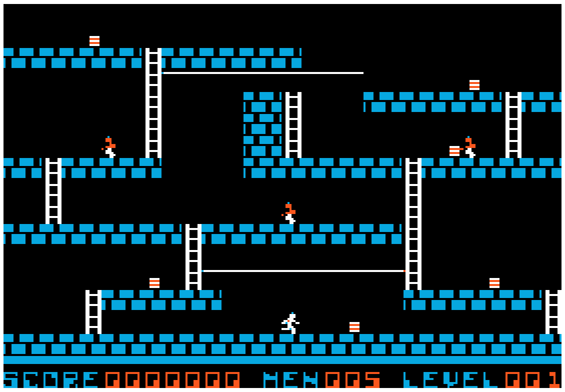
Sunday, January 22, 2023

Championship Lode Runner: Guard psychology and analysis of the C code

I expended a great deal of time working on a reckoning routine that would not be trapped by such lakes and yet would retain desirable economies of motion. After much wasted effort I discovered a better solution: delete U-shaped lakes from the map. - Chris Crawford



Anyone who's played Lode Runner for an appreciable amount of time comes to understand that the pathfinding AI can be a bit whack. It's clearly not a greedy algorithm, but it's also one that must take some shortcuts in order to fit in the Apple II+'s 48KB of memory, and it doesn't take much for the guards to get confused and march in erratic directions that get them nowhere. For the most part you don't really *need* to exploit or understand this - when the guards are acting dumb, getting stuck on ladders, and walking around in circles when there's a perfectly clear and direct path to getting the runner, you just shrug and grab those crates unmolested. Otherwise, you just find ways around them and dig traps for them to stumble into when they're in your way.

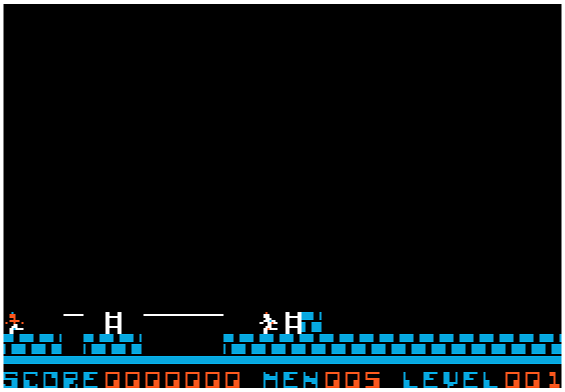
This is very different in Championship Lode Runner, a set of 50 user-made, Broderbund-curated levels whose creators, much like the authors of modern Kaizo-style ROM hacks, understood the engine better than its programmer had, and with this knowledge made bastard hard levels.

So, at the suggestion of *Renga in Blue* author and occasional commenter Jason Dyer, I thought I'd take a post to analyze the Lode Runner AI and better understand just what's going on inside those Bungelings' subpixel-sized brains. Coder Simon Hung remade Lode Runner, porting from C code that had been officially published in a Taiwanese reference book, and his conversion was invaluable in understanding this.

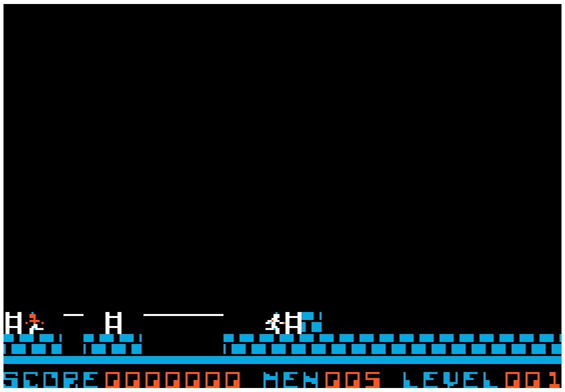
Goal 1: Get the runner

The first rule, after a few trivial rules that more concern physics than AI (e.g. falling, climbing out of pits, not passing through other guards, slowing down proportionally to the number of guards), is pretty simple to understand:

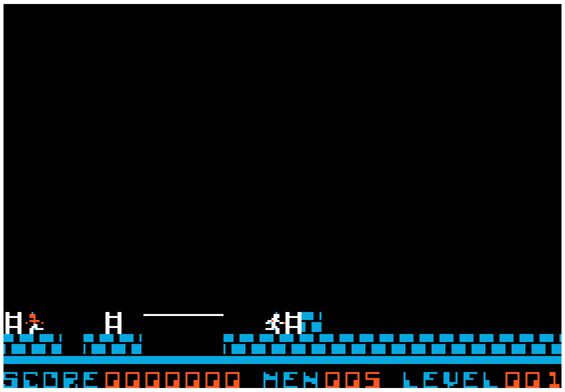
If I am on the same vertical level as the runner, and there is a clear path to the runner, then move toward the runner.



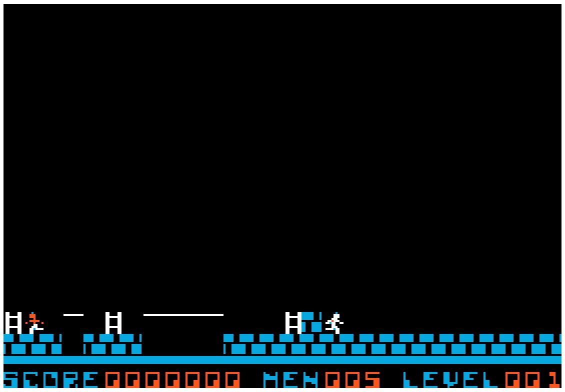
Pits dug by the runner are ignored, and subsequently fallen into.



Natural pits, however, are regarded as obstacles.



Walls, curiously, are also ignored. I'll assume I can just walk through them. I can't, but if the runner is on my level, I'll try anyway.



Climb the ladder? Are you nuts? That would move me *off* the runner's level!

Goal 2: Approach the runner's level

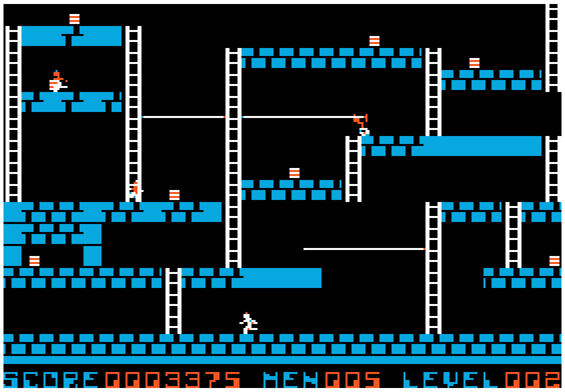
Here's where things get more complicated. If I can't just beeline for the runner, either because I'm not on the same vertical level, or because something seems to be in my way, then my second and final priority is to get to an advantageous level.

To do this, I will first look down, then up, then left, and then right. Each direction is assigned a score based on how advantageous it appears, and the lowest scoring direction wins. I will pick a direction that allows me to ascend or descend, but these are the outcomes from most to least favored:

- Get on the runner's level. In the event that there's more than one way to do this, the shortest path from me to the level (not necessarily to the runner himself!) wins.
- Get above the runner. If there's more than one way, then the closer I wind up to the runner's vertical level, the better.
- Get below the runner. The closer to the runner's vertical level, the better.

When multiple moves score the same, the first one examined wins.

Let's look at an actual level to illustrate:

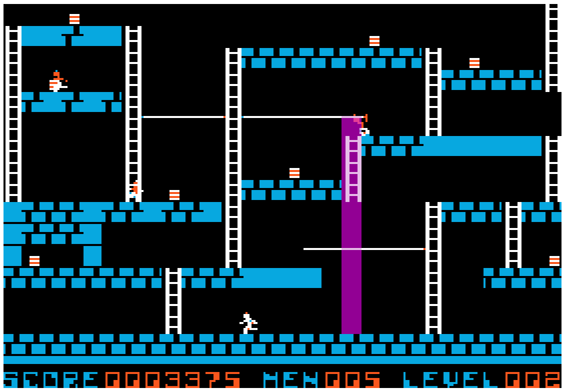


Three guards are present here, one on the upper-left, one toward the right, on the monkey bars, and another between them, by the ladder.

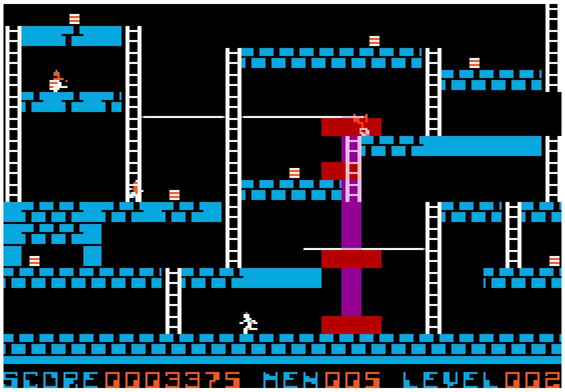
Step 1: Look down

Only the guard on the monkey bars has the option of moving down, so this step is meaningful only to him.

Guard logic: From above this ladder, I have a clear shot all the way to the bottom.

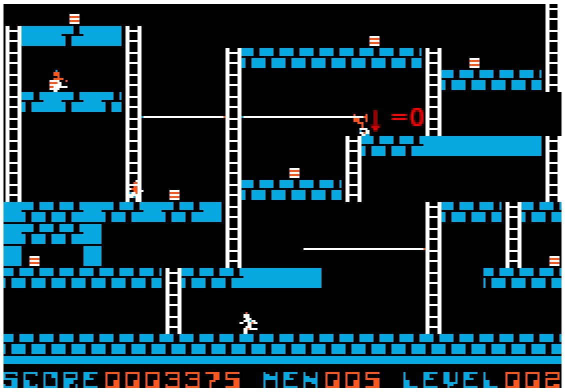


Let's look at all of the places in this column where I might disembark left or right:



I will **not** consider disembarking left or right from a ladder into a freefall unless I am at the top or bottom of it. The runner might consider it, but I won't.

What luck, the bottom of this drop, where I can move left or right, puts me exactly on the runner's level! This column is 0 tiles away from my current position, so I give this direction an unbeatable score of 0.

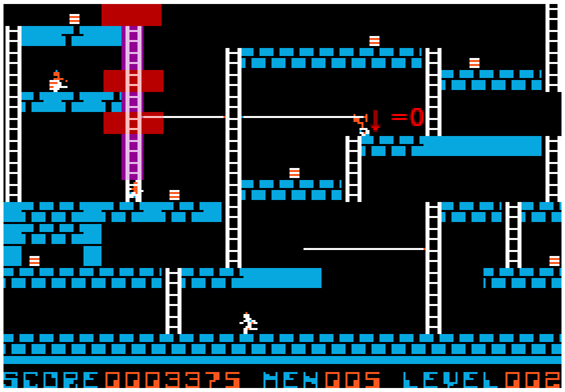


Notation: "↓ =0" : "If I go down I will be level with the runner with a score of 0."

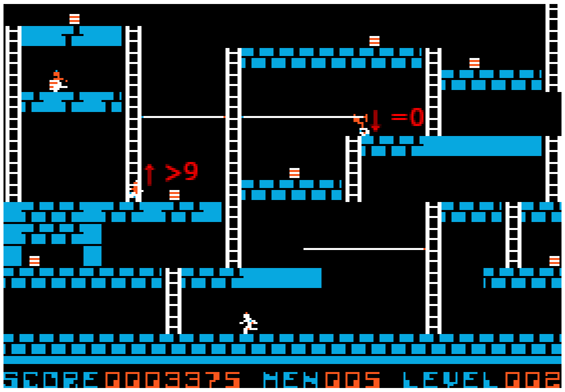
Step 2: Look up

This time, only the middle guard has the option of climbing upward, so we'll see about him:

Guard logic: I can climb this ladder to the top, and there are three places I might disembark from:



I'll already be above the runner at the first stop, so I must stop there and climb no higher. I'll be nine levels above the runner, so I give this path a score of nine.



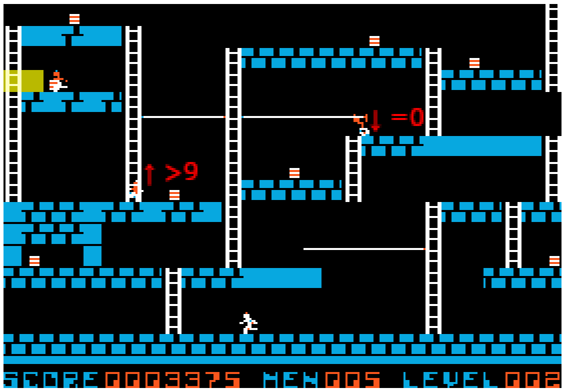
Notation: "↑ >9" : "If I go up I will be above the runner with a score of 9."

Step 3: Look left

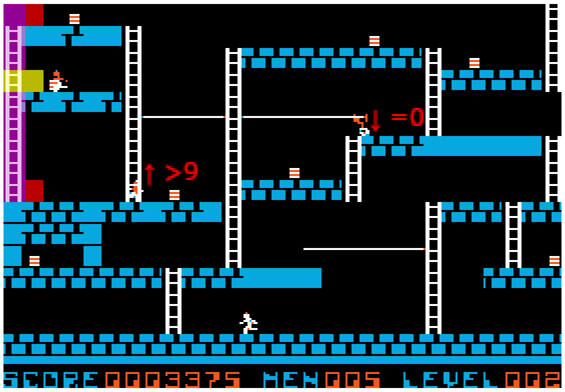
All of the guards can move left, so let's start with the leftmost one.

Left guard logic:

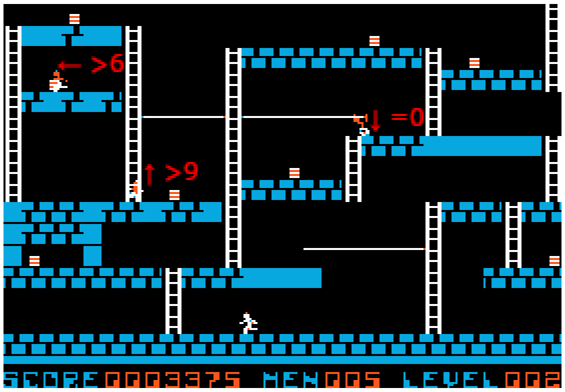
There's a ladder to my left.



There are two possible disembarkation points on this ladder:

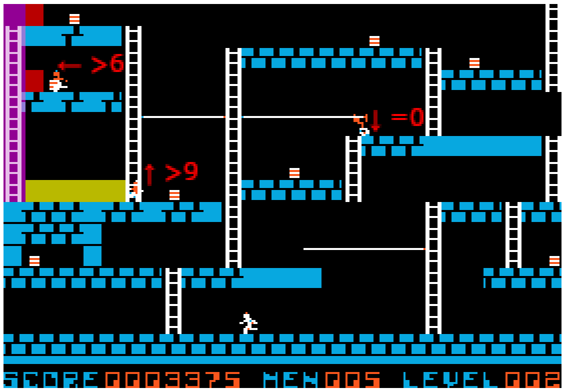


Both of these put me above the runner, but the bottom one is closer - six tiles above him, so I give this path a score of six.



Middle guard logic:

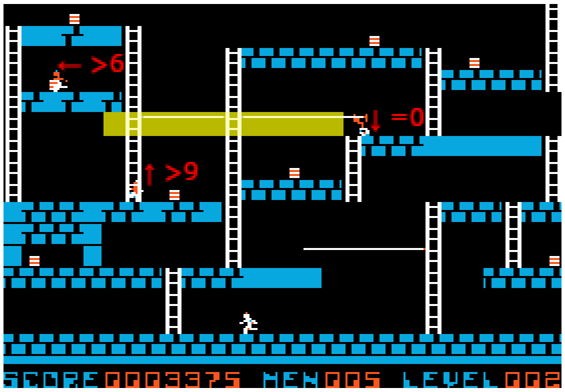
There's a ladder to my left, with two possible disembarkation points.



The closest of the two puts me 11 tiles above the runner. This is not better than going up, so I disregard this path.

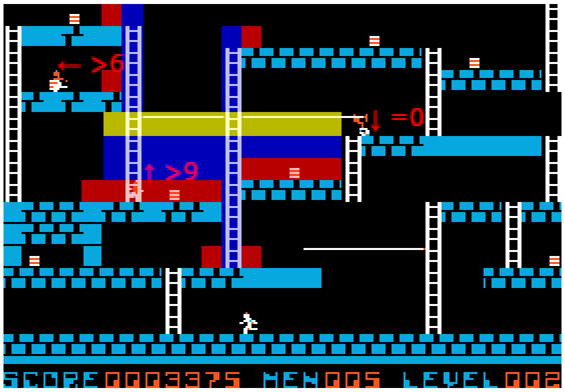
Right guard logic:

Let's see how far I can climb this bar to the left.



Note that, although I will not disembark into freefall from a ladder that I've climbed, this doesn't apply to ladders that I merely maneuver onto laterally.

That's a lot of options! Two ladders, each with multiple disembarkation points, plus I could just drop down from the bars where ever I feel like it!



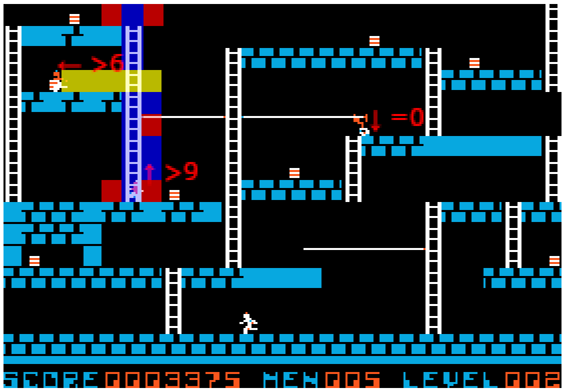
Climbing to the first ladder to my left and taking it down gets me the closest to the runner, but this doesn't beat the option of dropping down and landing exactly on his level. So I disregard this path.

Step 3: Look right

All of the guards can move right. Let's start with the leftmost one again.

Left guard logic:

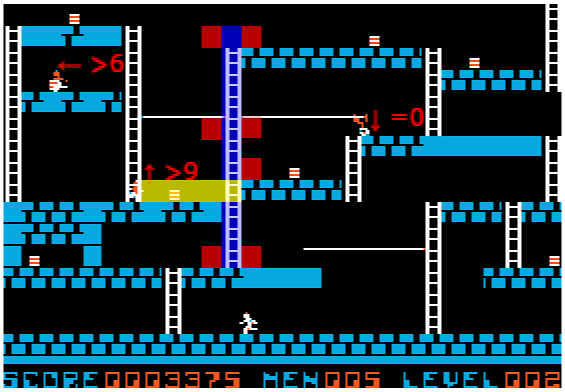
To my right, I can climb a ladder or keep going and drop off.



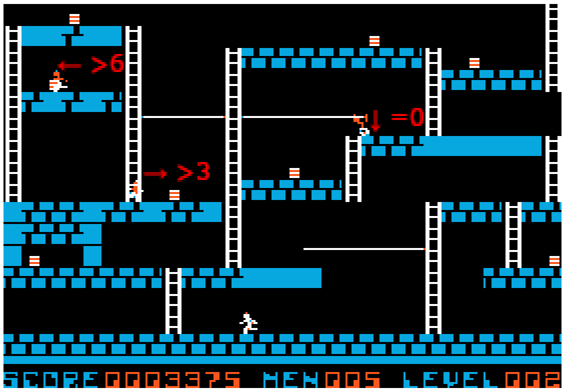
Either way, I'll land six tiles above the runner which does not beat the option of going left. So I disregard this path.

Middle guard logic:

There's a ladder to my right, which I could climb up or down.

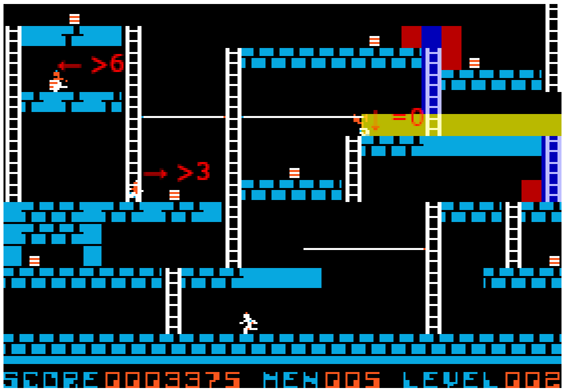


Down all the way still puts me above the runner, but closer than my previous score (3 vs. 9), so this becomes my new path.



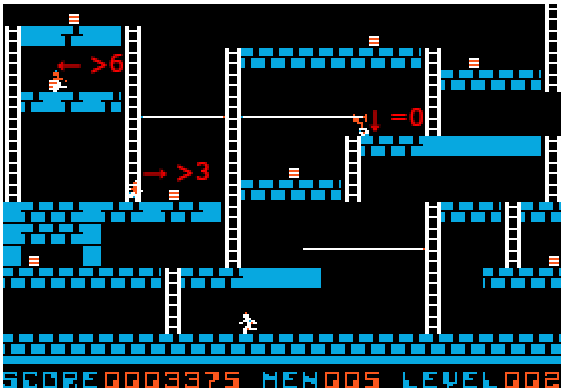
Right guard logic:

To my right I can either climb up a ladder or down a ladder.



Neither option is as good as my initial path of just dropping down to the bottom, so I disregard this path.

The final decisions:



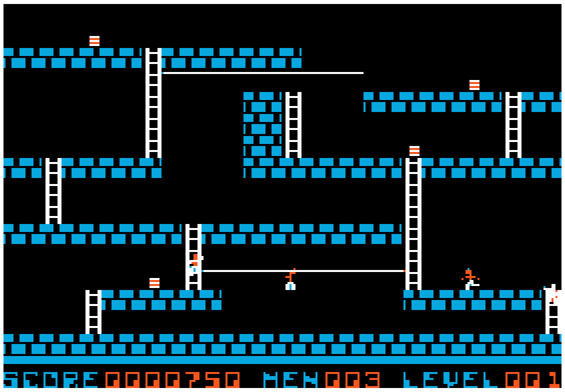
Left guard goes left, so he can climb down the ladder and get you. Middle guard goes right, so he can climb down the ladder and get you. Right guard drops down so he can get you.

Seems elegant, right? Well,

When things get weird

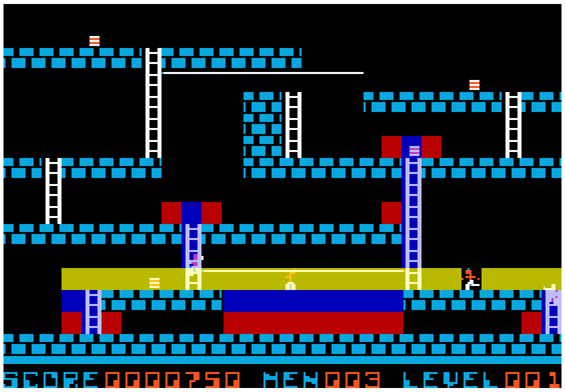
Let's look at some examples of ways this two-step logic can go very wrong, which Lode Runner: Championship Edition demands you understand well enough to invoke on purpose:

Example 1: Fleeing the target



I'm on the ladder in the lower-right corner. So why are the guards running away? Why not just go right, climb down a tile, and kill me?

Simple - the AI doesn't realize it can reach this spot! As far as pathfinding goes, it is only considering places it can disembark from ladders and/or drop from.

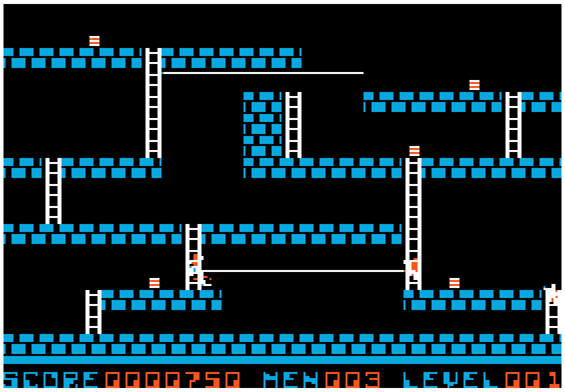


The runner's spot was never considered a reachable endpoint. The spot below him to the left is, and the path there does cross the runner, so why not go there?

Because getting above the runner is better than getting below him. Doesn't matter how much farther away you wind up. Better to be three floors above the runner than one floor below him.

And so, the guards all go to the left, where they can climb ladders to a spot three floors above the runner. But what happens when they reach the ladders?

Example 2: Ladder physics suck



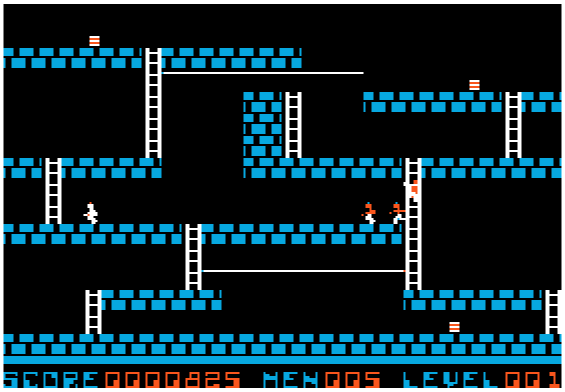
Ladder trouble - it's not just for Gordon Freeman

So the guard decided it's going to go for the ladder and take it up to a higher vantage point, right? But then when it gets to the ladder, it just sort of gets stuck there.

The reason is simple. When the guard is at the base of the ladder, it wants to climb up to the nearest disembarkation point. But the guards recalculate their paths with every step they take, and once the guard has climbed and is no longer considered to be at the base, the base becomes a valid disembarkation point, so he climbs down, entering an infinite loop! The guards on the left ladder are attempting the same, but have created a log jam.

If you climb your ladder up or down, you will become level with a valid disembarkation point and the guards will unstuck themselves.

Example 3: A reverse false wall



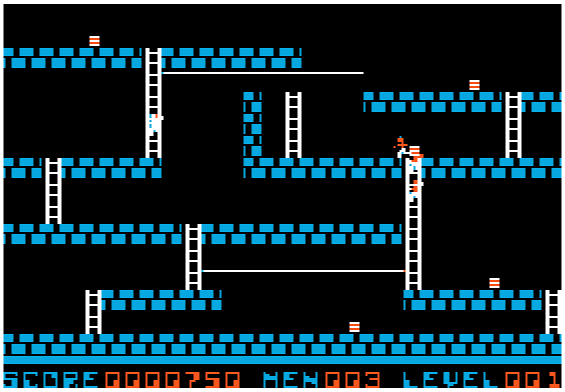
Two weird things are happening here.

First, as soon as you climb the first ladder, the guards chasing you turn around and climb the other ladder. They do this because both ladders will get them to your level, but the ladder on the right is closer (to them).

Second, once the guards reach the second ladder, they just won't go over the top of the ladder, as long as you don't. What gives? Shouldn't they want to reach the top, so they can reach the bars just above you, climb over, and drop down?

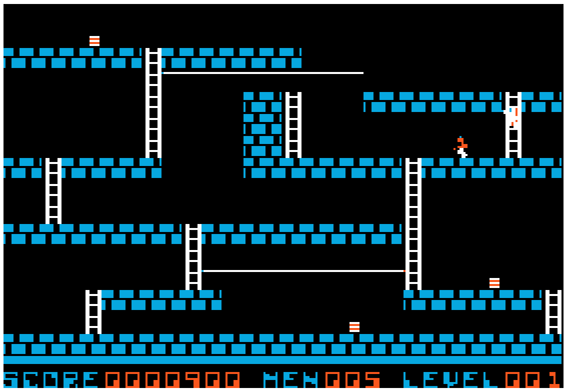
They would, except for one thing. Ladders against walls are confusing to guards, because their logic for determining valid disembarkation points determines walls in relation to where their feet will be, but not where their bodies will be! Every tile on the ladder is treated as a valid disembarkation point for pathfinding purposes - as long as you don't rise above the ladder, they'll *try* to beeline for you to the left, thinking they can step on the tile at their feet, but they can't because there's a tile in their way. They get stuck there.

Remove the wall, and they act a bit more sensibly.



Emphasis on a bit.

Example 4: Left-to-right reading order



Before I climbed that ladder, the guard was chasing me to the right. After I started climbing it, he turned around and started running for the ladder to the left. We already examined why he wouldn't necessarily chase me up the ladder, but why turn around and go for the one that's farther away?

It's because neither ladder reaches me (as far as the AI is concerned), but both go to the same level above me. Left gets evaluated before right, so left wins.

This is far from a comprehensive guide on the guard AI, and does not cover the many corner cases both implicit and explicitly defined in the game logic, nor any of the peculiarities of the game physics that sometimes dictates their behavior, but I think this more or less explains how the Lode Runner guards think, and illustrates what's going on in most of their screw-up situations. For now, I'm going back to Championship Lode Runner's punishing levels.

Labels: Broderbund

7 comments:

The Wargaming Scribe January 22, 2023 at 5:28 PM

Really interesting ! As a kid I "got" most of the AI behavior, but only by pattern recognition, not by understanding the code below it. It makes a lot more sense, now.

Reply

Jason Dyer January 22, 2023 at 9:01 PM

>They would, except for one thing. Ladders against walls are confusing to guards, because their logic for determining valid disembarkation points determines walls in relation to where their feet will be, but not where their bodies will be! Every tile on the ladder is treated as a

valid disembarkation point for pathfinding purposes - as long as you don't rise above the ladder, they'll try to beeline for you to the left, thinking they can step on the tile at their feet, but they can't because there's a tile in their way. They get stuck there.

This is amazing.

I admit this is my favorite kind of post of yours, just because so few people take the time to get at this level of detail!

Reply

Replies

The Wargaming Scribe January 23, 2023 at 4:44 PM

I agree that those analytical posts of yours are amazing !

Reply

MalcolmM January 23, 2023 at 1:31 AM

As a retired programmer who always wanted to work on computer games but never managed to, this analysis is fascinating to me. I completed all the levels in the original Lode Runner, admittedly using emulator save states. I realized that at times the AI was a bit wonky but your analysis makes me realize some of the many factors it has to take into account. Pretty impressive for 48k.

Reply

Harland January 25, 2023 at 5:14 AM

About time we got back to a more data-driven perspective. Honestly I could do with a whole series of these posts with animation and examples from play. There was a Youtube video about the ghost logic in Pacman where the video explained the rules and then displayed where the ghosts were thinking about going as the game was played. It was outstanding. Here, I've found it: https://www.youtube.com/watch?v=ataGotQ7ir8

Reply

Replies

Viila January 27, 2023 at 3:28 PM

I concur, I really enjoyed this episode of DDG. It was interesting dive into the behavioral logic of the guards. As a developer myself, I appreciate the simplicity of the rules and how they still create quite clever behaviour. If there are some things I might have changed myself to improve it, is to add some hysteresis/memory so that the guards don't flip every frame to a new goal like that. Perhaps so if they start on ladder, they commit to their exit point before calculating a new path.

Reply

ERoberts January 28, 2023 at 1:13 PM

This is the good stuff. Thank you Ahab.

Reply

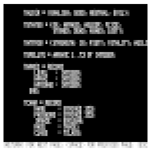
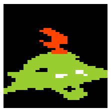
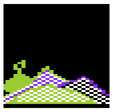
Enter Comment

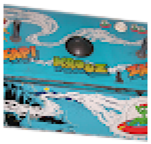
Commenting with signin or name/URL is encouraged but not required. If the spam filter deletes your legitimate comment, apologies - it does that sometimes.

Newer Post Home Older Post

Subscribe to: Post Comments (Atom)

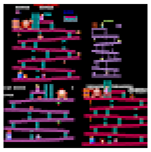
Most popular posts

- 
  The not-so-basic mechanics of Wizardry
  Around 2012-2014, Thomas William Ewers reverse-engineered the Wizardry code and published compilable Pascal source to Asimov, which was...
-
  Game 85: Wizardry: Proving Grounds of the Mad Overlord
  I've hacked together a "fixed" version of Wizardry for the Apple II which restores the initial game state to what I believe is the correct o...
- 
  The bestiary of Wizardry
  I've split the monster table into several, while also merging some of the columns. Trust me, this is the least bad solution that I...
- 
  The treasury of Wizardry
  In my last post, I briefly described the treasure system of Wizardry, where one of two treasure values are selected to determine your re...
- Unusual MAME controls
  Arcade games had a wide variety of controls. Most of the time, they translate pretty well to a gamepad – by far the most common coin-up con...

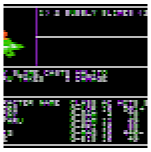


- Versions of Zork I and their text differences

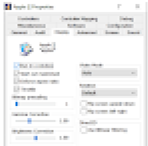
  For my last posting on Zork I, I want to highlight the differences between versions. It’s something I wanted to know about, and I couldn’t f...

-  The Famicom's first games

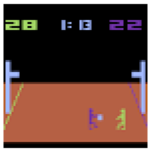
  July 15th, 1983 marked the domestic launch of the most important video game system in history. It was Japan's first truly successful ho...

-  Wizardry: Proving Grounds version 3.0

  Version 2.1 of Wizardry for the Apple II came out just over 41 years ago, and even though it's been ported and remade a dozen times ov...

-  On Apple II emulation by MAME

  I’ve always used AppleWin to emulate Apple II games. Back in 1996, it came bundled with the Roberta Williams Anthology as a means to play ga...

-  Game 280: One on One - Dr. J vs. Larry Bird

  The first sports game I ever covered was Atari's Basketball from 1978. A simplistic one-on-one basketball game, it nevertheless antici...

Simple theme. Powered by Blogger.