Discrete Hit Detection:

Normal bounding rectangle hit detection tests that Grondin teaches has a few problems when you try to use it in a platformer game. One of them is that when the player jumps and hits a block you don’t know that the player has hit anything until it is stuck in it. You then need to figure out how to get the player unstuck. You also need to treat hitting sides differently than hitting the top and you get stuck with awkward corner cases that are nightmares to solve.

Discrete hit detection uses the same basic overlapping rectangle idea but prevents the player from ever actually hitting a block. Discrete detection instead moves the player as close as possible to a block without actually touching it.

The way I implemented discrete hit detection is to “pretend” to move the player and if they are hitting anything to pull them back and “pretend” move them a little bit less than you tried to last time. Once they are in a spot where they don’t hit anything you can actually move them there. To do this, I made two recursive functions for the X and the Y that will get the player as close as possible to the blocks without touching them. I loop through calling the functions that will pull back on the X and Y until the player finds a spot it can move to.

When I say that you need to “pretend” to move the player around I mean you have a function that returns a rectangle at the spot where the player would be if you actually moved them but don’t actually change the sprites real position.

<https://katyscode.wordpress.com/2013/01/18/2d-platform-games-collision-detection-for-dummies/> This blog has a lot of good information on hit detection. The author explains the basics of many types of hit detection and it is where I learned about discrete hit detection. Her example code implements a slightly more complicated version of hit detection that checks individual points on the player against the terrain but it was not needed for my game so I never went into it.