I have a feeling that now you are starting to realize where some of those things you learned in your math classes are actually used by people in the real world.

As we just saw, the cosine and sine properties are very popular among game programmers. You can use these tools every time you need to work with angles or rotation.

Just a recap from before. Remember… “sine” gives us the ratio between the opposite side and the hypotenuse, and “cosine” gives us the ratio between the adjacent side and the hypotenuse.

That means that, if we have either one of the sides and the hypotenuse, we can get the sine or the cosine of the angle.

…

But, what if we don’t have the value of the hypotenuse and we only have the values of the two sides?

Is there a property that gives me the ratio between the opposite side and the adjacent side?

Yes, there is. This is called the tangent of an angle.

The tangent of a given angle gives us the ratio between the opposite side and the adjacent side.

This ratio between the opposite (y) side, and the adjacent (x) side, is sometimes called “rise over run.”

People call this the “rise over run” because it gives us the ratio between how much something is changing in the up/down Y-axis (the rise), over how much it is changing in the left/right X-axis (the run).

Now you know that, if you have an angle, you can find the sine, cosine, and its tangent using Lua’s built-in functions math.sin(), math.cos(), and math.tan().

This is all beautiful and useful. But sometimes, when we are programming a game, we need to find things the other way around.

What if we have the ratio of the sides and we need to find the actual angle that is formed by the sides?

That is why we also have what we call the inverse trigonometric properties.

Do you remember reading about arc-cos, arc-sin, and arc-tan?

There you go… they are the inverse functions of cos, sin, and tan.

Let’s take a look at the arc-tangent, since it’s very popular between game programmers.

Think of arctan function like a machine that finds angles… in one side you put the two sides, and the output of the machine is the angle that is formed between those two sides.

<machine-image>

This is incredibly useful for us, since we can find the angle of an object in the screen if we know the position x and y of the game object.

As always, Lua comes to the rescue. The Lua language has a built-in function to compute the arctan.

math.atan(opposite / adjacent)

math.atan() receives the ratio between the opposite side over the adjacent side, and gives us back the value of the angle formed by the sides.

And remember, whenever we talk angles in Lua, we are talking angles in radians.

So, instead of 0 to 360, Lua’s atan() function will give us back angles from 0 to 2\*PI (approximately 6.28 radians).

…

Alright, let’s see if all of these ideas make sense in the real world.

The next project has two game objects in the screen. A red target, and a small white bullet projectile, that is currently being shot from the origin to the right, with an angle of 0.

Your goal is to change the code to make the game shoot the bullet in the direction of the target.

In other words, you will find the angle produced by the position x and y of the target, and use it as the angle for the bullet movement.

Remember the arctan property, that receives the ratio of the opposite and adjacent sides of a triangle and gives back the angle formed between them. I’m pretty sure that will help you in your task.

The good news is this is your last project of the day. Once you’re done, just type “exit” in the terminal and you can go home to watch some Netflix.

I have to say I am having fun… I hope you are too.

Alright, see you soon.