Compsci.373 Tutorial 7

COMPUTER GRAPHICS III

Today’s outline

- A look at the Assignment skeleton code
- The rasterization process
- A few exercises

The Assignment skeleton code

Your assignment is almost up!

The resources of the programming assignment will consist of two parts:

1. A skeleton project for you to complete (if you choose to)
2. A document detailing the skeleton project, and what steps you need to take to complete it.

The Assignment skeleton code

Because the assignment is assessed entirely online, this limits the complexity of the questions we can ask you there. That means that you will have to put more effort in to complete the skeleton, but there are plenty of reasons to do it:

- You'll be able to complete the online assignments by just pasting your code.
- You'll gain a much better understanding of Ray Casting for the exam.
- You'll have a working Ray Casting engine!

Lets look at the skeleton code

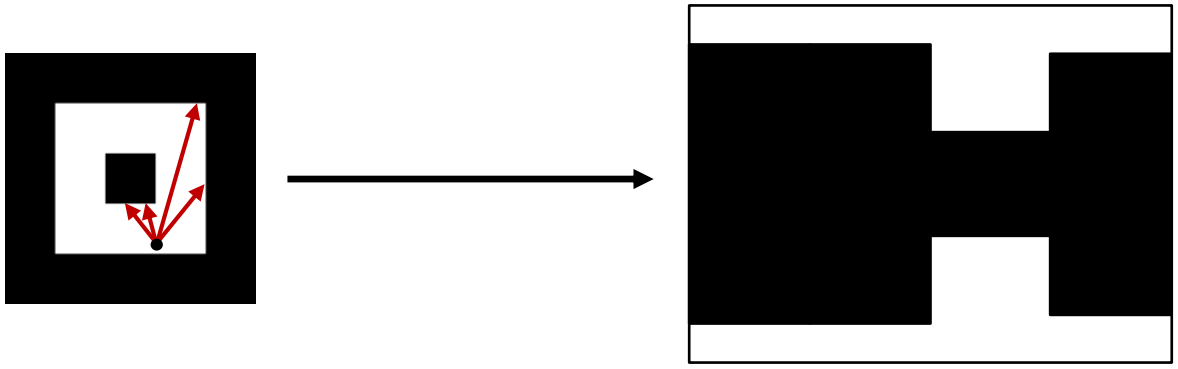
A quick recap on Ray Casting

Recall the Ray Casting procedure:

1. Define our rays
2. Normalize our rays
3. Find the step sizes for our rays
4. Step the rays into the world until they hit something

A quick recap on Ray Casting

Once the rays are all hitting something in the world, we have all that we need to begin the rendering process.



What we have

What we want

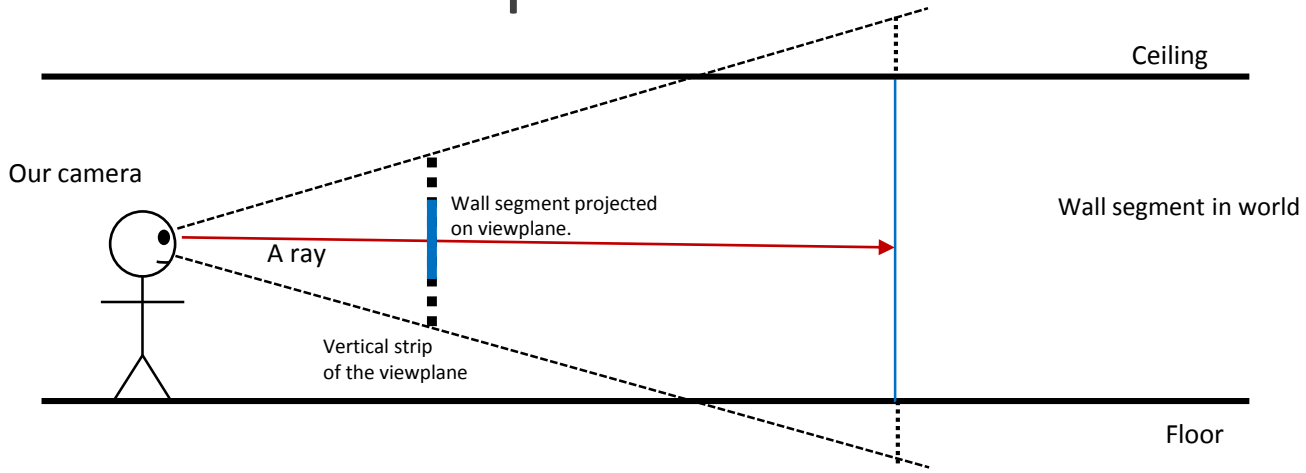
The rasterization process

We want to draw a vertical column of pixels for each ray, where the height of the column depends on how far the corresponding ray needed to travel.

The relationship between the height of the pixel column we want to draw, and the length of the corresponding ray is inversely proportional. That means that the further the ray had to go, the smaller our pixel column.

The rasterization process

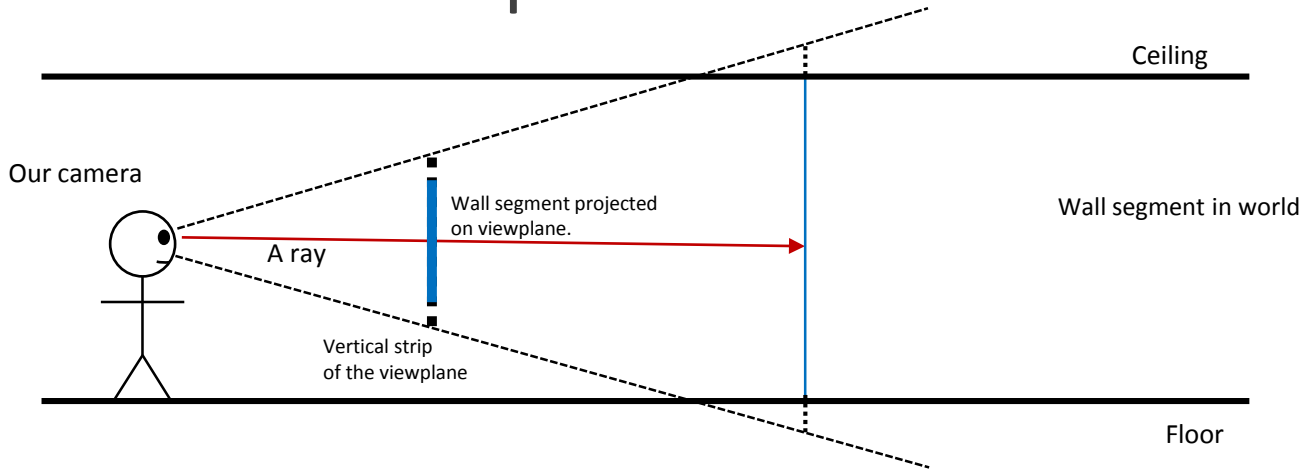
We need to define this relationship, but first, it helps to take another view of the problem.



This is the view of one ray from the side.

The rasterization process

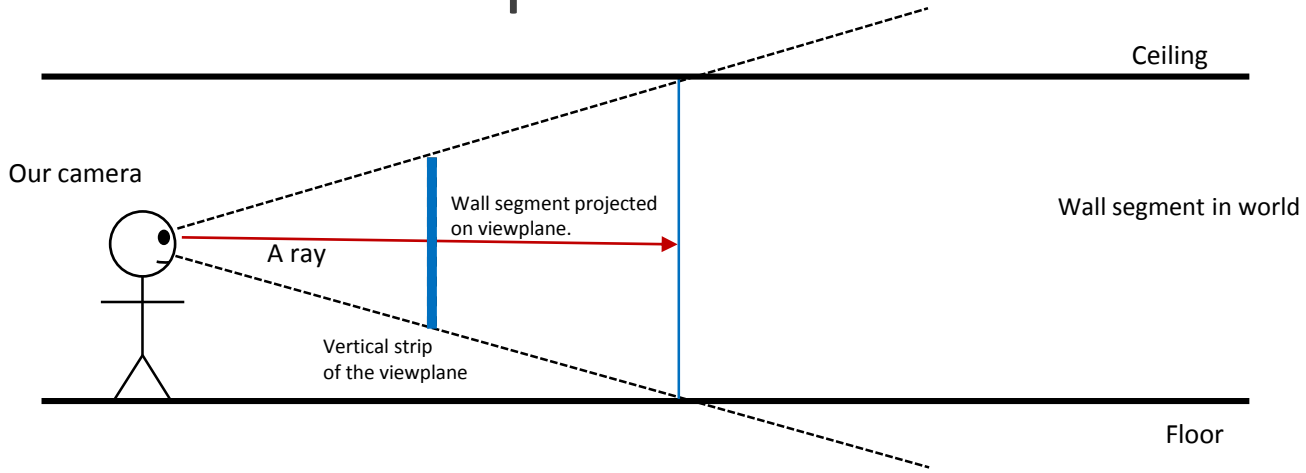
We need to define this relationship, but first, it helps to take another view of the problem.



This is the view of one ray from the side.

The rasterization process

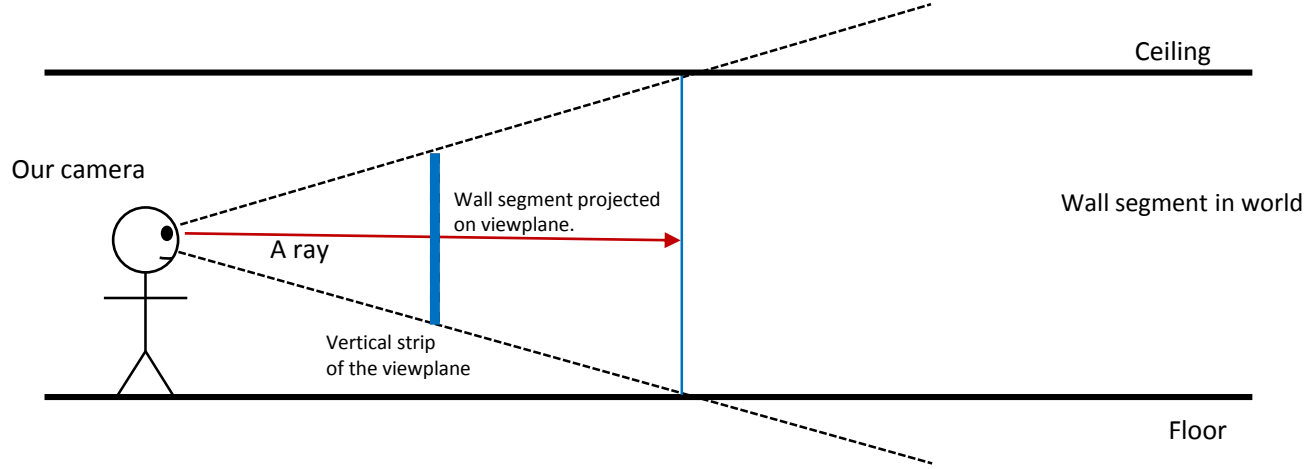
We need to define this relationship, but first, it helps to take another view of the problem.



This is the view of one ray from the side.

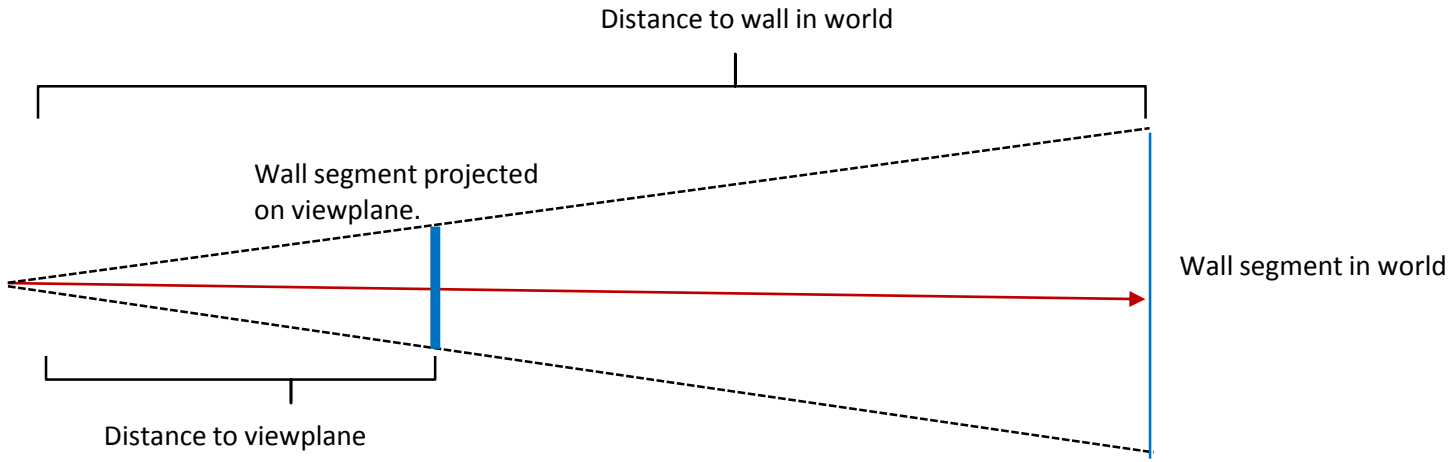
The rasterization process

Notice how the strip on the viewplane gets larger the closer the wall is? How are the sizes related?



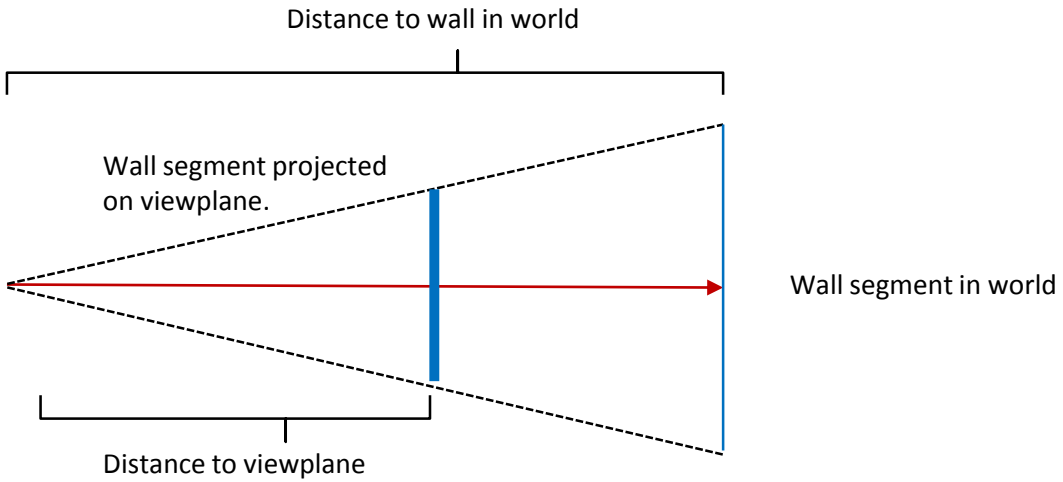
The rasterization process

The following triangle is formed from the distance to the viewplane and the height of the drawn pixel column, and the distance to the wall and the height of the wall.



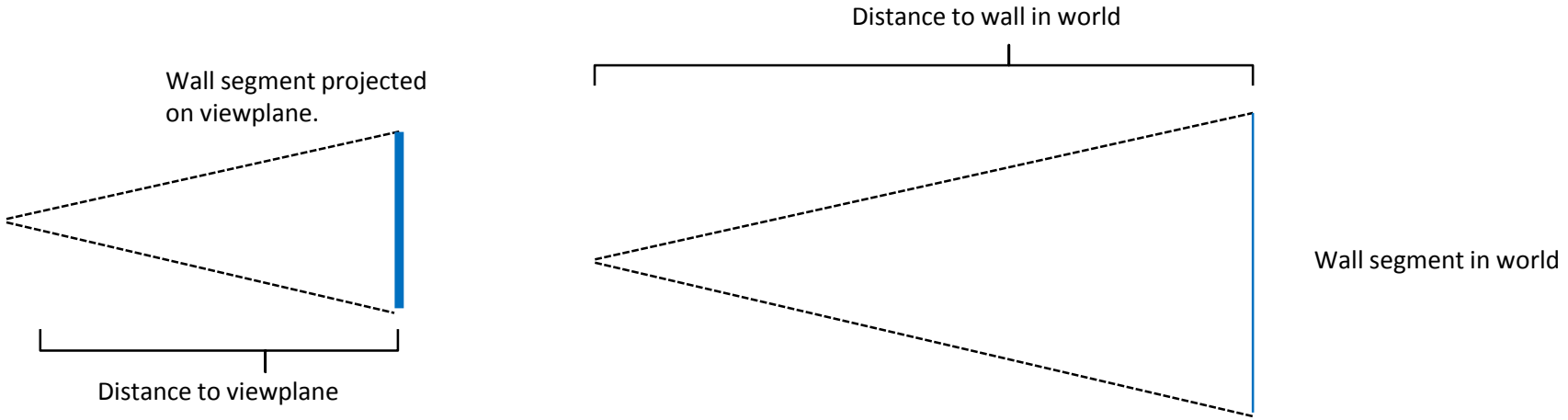
The rasterization process

The following triangle is formed from the distance to the viewplane and the height of the drawn pixel column, and the distance to the wall and the height of the wall.



The rasterization process

This is great, because we can leverage the properties of similar triangles to form a relationship between the wall sizes. We essentially have two triangles which are different scales of each other:



Because these are scaled versions of the same triangle, they both have the same ratio between the length of the far side of the triangle, and the length of the triangle.

The rasterization process

This gives us the relationship:

$$\frac{\text{Projected wall height}}{\text{Distance to viewplane}} = \frac{\text{Actual wall height}}{\text{Distance to wall in world}}$$

Which rearranges to:

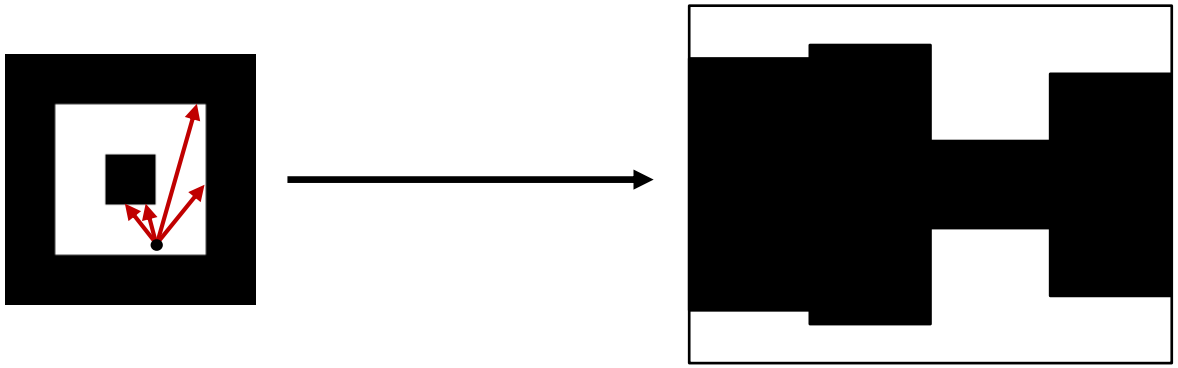
$$\text{Projected wall height} = \frac{\text{Actual wall height}}{\text{Distance to wall in world}} * \text{Distance to viewplane}$$

Giving us:

$$\text{Projected wall height} = v_d \frac{\mathbf{S}}{|\boldsymbol{R}|}$$

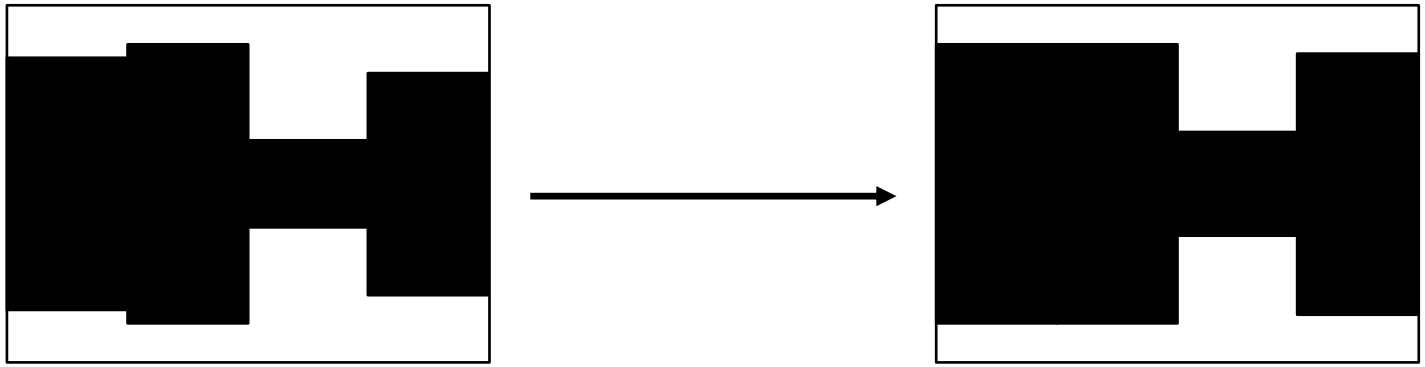
The rasterization process

Now that we have that relationship, we can draw one column per ray:



The rasterization process

Hold on, that looks distorted!

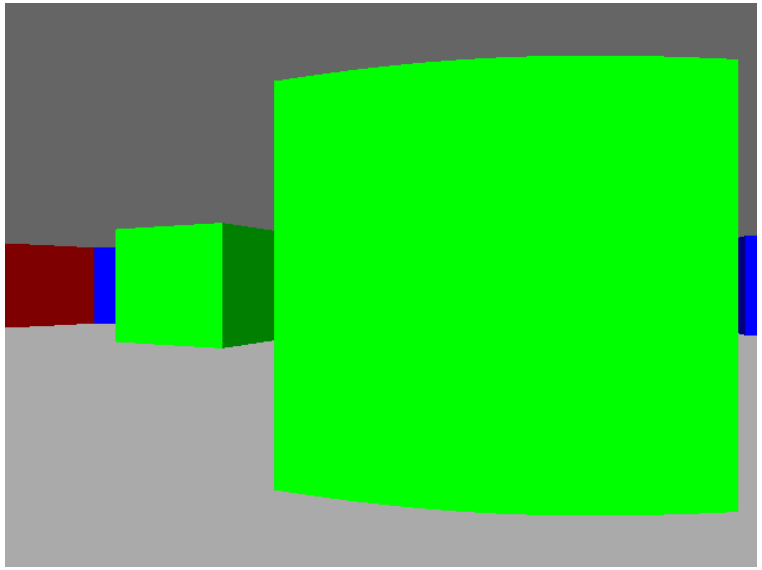


What we have

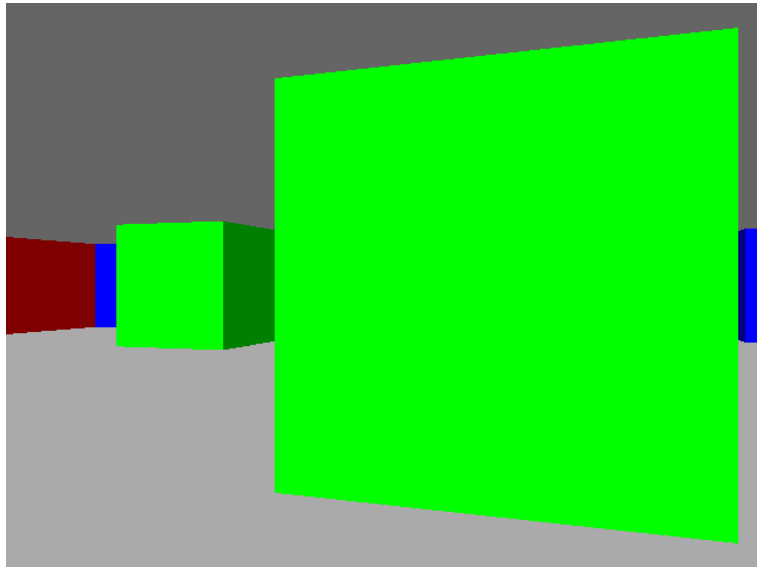
What we want

The rasterization process

Hold on, that looks distorted!



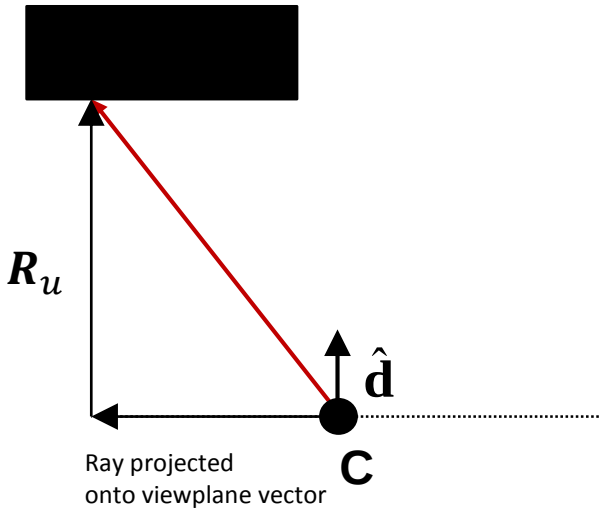
What we have so far



What we want

The rasterization process

To calculate the correct distance, we simply take the orthogonal vector projection of our ray onto our viewplane vector and finally subtract this vector from our ray for our undistorted ray vector, which we will call $\boldsymbol{R}_u$. Using projection, this gives us:



$$\boldsymbol{R}_u = \boldsymbol{R} - \widehat{\boldsymbol{V}}\frac{\boldsymbol{R}\cdot\widehat{\boldsymbol{V}}}{\widehat{\boldsymbol{V}}\cdot\widehat{\boldsymbol{V}}}$$

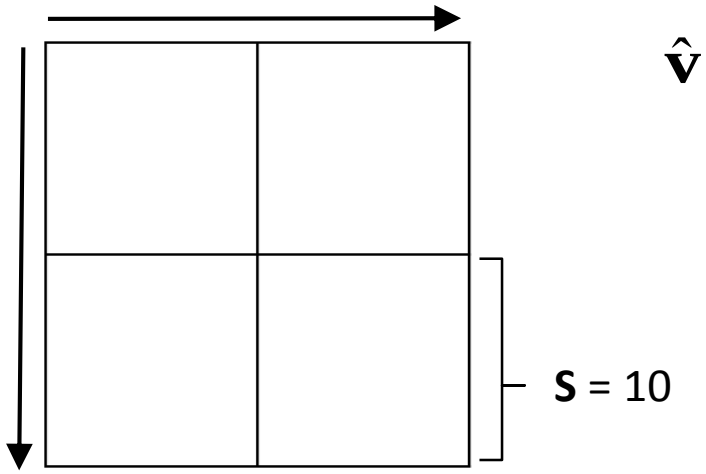
Or

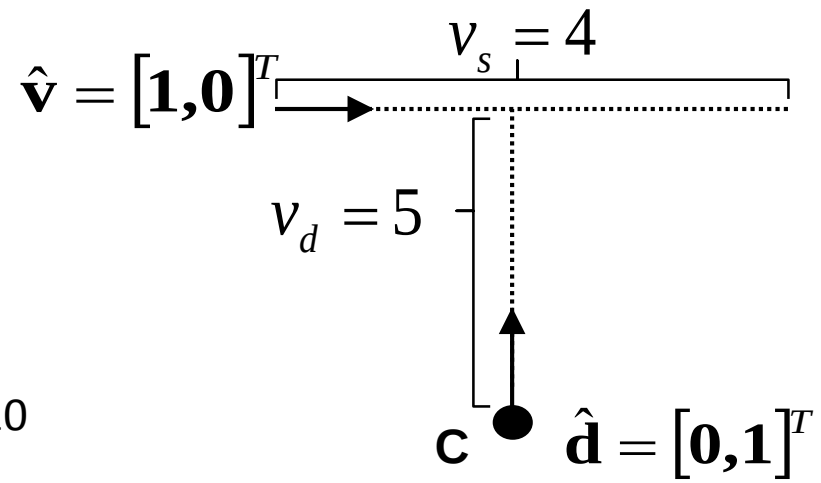
$$\boldsymbol{R}_u = \boldsymbol{R} - \widehat{\boldsymbol{V}}(\boldsymbol{R}\cdot\widehat{\boldsymbol{V}})$$

(because $\widehat{\boldsymbol{V}}$ is normalized)

Exercises

Assume the following full Ray Casting environment:





$$\mathbf{c} = [\mathbf{14,17}]^T$$

Exercises

1. What is the normalized direction of the 2nd ray (n = 1)?
2. What is the initial vertical step vector, $\boldsymbol{V_I}$?
3. What is the initial horizontal step vector, $\boldsymbol{H_I}$?
4. What is the main vertical step vector, $\boldsymbol{V_S}$?
5. What is the main horizontal step vector, $\boldsymbol{H_S}$?
6. What will the 2nd ray be when it has been stepped to the 3rd vertical intersection?
7. What will the 2nd ray be when it has been stepped to the 3rd horizontal intersection?
8. What is the undistorted length of these two rays?

That’s all for today!