Finding stronghold location with triangulation

diegoberti#7582 —— @diegoberti

Updated to 16 August 2024

1 Introduction

The method exploited by this tool is as simple as it is effective. However, often performing calculations of trigonometry, specifically, triangulation, can be really complex if you do not know some basic properties of mathematics; that is why using this tool **saves considerable time**.

2 Credits

I think it's important to thank those from whom I took inspiration:

- Minecraft Locating the Stronghold - A Speedrunner's Guide

3 Glossary

- **Tool**: The website I developed — StrongholdFinder
- **Distance**: The preset distance of the Tool — 17.5 blocks
- P_1[1]: Player's first position
- P_2: Player's second position
- P_3: Approximated stronghold location
- θ_1[2]: Angle on which the line starting from P_1 lies
- θ_2: Angle on which the line starting from P_2 lies

[1] Every P_n are sets of (x_{P_n}, z_{P_n}) coordinates
[2] Every θ_n is the angle of a Eye of the Ender throw

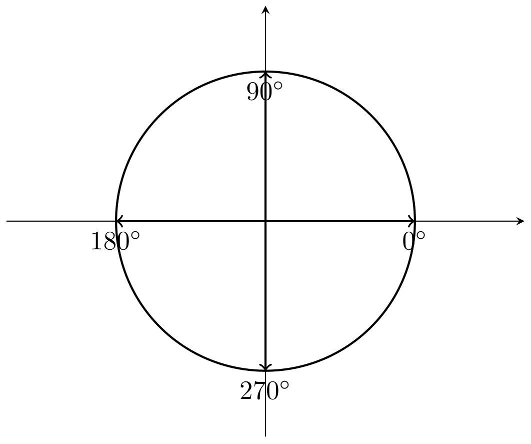
4 How Ender Eyes works in Minecraft

The Eyes of the End, once thrown by the player, follow the **fastest trajectory to the chunk where the stronghold is located**, heedless of the obstacles in front of the player. Usually, the player understands the trajectory and continues to follow it until the said chunk is reached, casting more Eyes of the End from time to time.

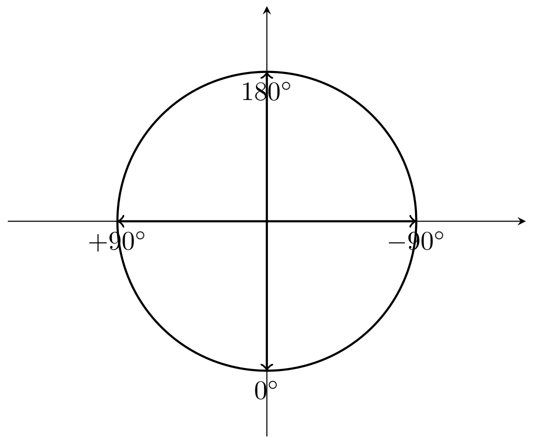
There is one exception: in the rare case where the player is almost in the middle of the distance between two different strongholds (i.e. 999 blocks and 1001 blocks away), the Eye of the End thrown will aim at the one closest to the player. In such a case, if the player does not correctly understand the trajectory of the eye, he or she may end up in the radius of another stronghold, **greatly increasing the distance to be traveled**.

5 How the angles works in Minecraft

We are used to knowing angles (in degrees) as a circumference whose values range between 0° and 360°.



In Minecraft this **is not quite so**: although the absolute sum of the values is 360°, they work **differently**.
In fact, in Minecraft, the angle of value 0° is placed toward the **South, toward the positive Zs**, growing by turning toward the East until it reaches 180°. Angles greater than 180° take on a **negative value by turning toward the West**, decreasing to 0°, thus completing one complete turn.



6 Calculations and operation of StrongholdFinder

The calculations are actually quite simple: everything is based on the fact that there is a ray half-line defined when it starts from a point P_1 and is adjacent on an angle θ_1.

At this point it is simple: by finding the straight line R_1 that begins at P_1 and angle θ_1, and another straight line R_2 that begins at P_2 and angle θ_2, we form two cathexes of a triangle. The basis of this is represented by the distance P1-P2, which in the tool is fixed at 17.5, and the only thing left for us to do once we find the straight lines is to find the intersection of these at P_3, which in our case is the position of the stronghold.

However, having a triangle with random vertices, and therefore angles, would make the calculation:

- Complex to accomplish
- Inaccurate

This is precisely why once the position P_1 is recorded, the tool requires the player to move 17.5 blocks by turning 90°, before doing another throw. The nice thing is that the calculation is done automatically through the trivial use of the following formulas:

6.1 Second throw X coordinates generation

- First of all, if the angle is negative, it should be subtracted from 360:

$$\theta_1 = \begin{cases} \theta_1 & \text{if } \theta_1 > 0 \\ 360 - |\theta_1| & \text{if } \theta_1 \leq 0 \end{cases}$$

  The angle must be converted in radians, that's why it's present:

$$\frac{\theta_1 \cdot \pi}{180}$$

- Now we can explain the full formula:

$$X_{coordinates} = x_{P_1} + 17.5 \cdot \cos\left(\frac{\theta_1 \cdot \pi}{180}\right)$$

6.2 Second throw Z coordinates generation

- The process is very similar to the X coordinates generation. First of all, if the angle is negative, it should be subtracted from 360:

$$\theta_1 = \begin{cases} \theta_1 & \text{if } \theta_1 > 0 \\ 360 - |\theta_1| & \text{if } \theta_1 \leq 0 \end{cases}$$

- Now we can explain the full formula:

$$Z_{coordinates} = z_{P_1} + 17.5 \cdot \cos\left(\frac{\theta_1 \cdot \pi}{180}\right)$$

6.3 Generated coordinates set

These formulas returns a coordinates set for the P_2 position.

6.4 Second throw

The player is required by the tool to throw another Eye of the Ender.

The tool now performs an absolute subtraction between θ_1 and θ_2

$$\Delta_\theta = |\theta_2 - \theta_1|$$

6.5 Stronghold X coordinates generation

- First, we apply the same analysis of the second angle as explained before in *6.1*

$$\theta_2 = \begin{cases} \theta_2 & \text{if } \theta_2 > 0 \\ 360 - |\theta_2| & \text{if } \theta_2 \leq 0 \end{cases}$$

- Then we can apply the full formula:

$$x_{stronghold} = x_{P_2} + \left(\frac{17.5}{\sin\left(\frac{\Delta_\theta \cdot \pi}{180}\right)}\right) \cdot \cos\left(\frac{(\theta_2 + 90) \cdot \pi}{180}\right)$$

6.6 Stronghold Z coordinates generation

- First, we apply the same analysis of the second angle as explained before in *6.1*

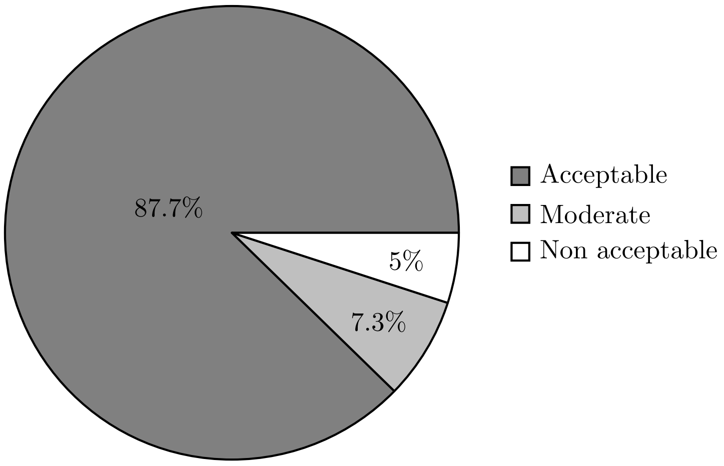
$$\theta_2 = \begin{cases} \theta_2 & \text{if } \theta_2 > 0 \\ 360 - |\theta_2| & \text{if } \theta_2 \leq 0 \end{cases}$$

- Then we can apply the full formula:

$$z_{stronghold} = z_{P_2} + \left(\frac{17.5}{\sin\left(\frac{\Delta_\theta \cdot \pi}{180}\right)} \right) \cdot \sin\left(\frac{(\theta_2 + 90) \cdot \pi}{180} \right)$$

6.7 Generated stronghold coordinates set

These formulas returns an approximated set of coordinates $(x_{stronghold}, z_{stronghold})$ of the stronghold chunk with a precision of the 87-88%.



7 Notes

Although I tried to make this tool as precise as possible, it might get wrong position. You must **NOT** rely on these and you **ALWAYS** have to double check them.

I don't want responsibility because you didn't double-check the coordinates. Thank you.