Predicting stronghold locations with Bayesian statistics

Discord user Ninjabrain#9740

November 2020

1 Introduction

The traditional triangulation technique involves throwing two eyes of ender and intersecting the lines they trace. This traditional model will work perfectly every time if the user is able to read the angle of the ender eye perfectly. However, this is not possible since the user has to point their crosshair manually at the eye, adding a measurement error. In this paper I will present a model that accounts for the error, and has multiple benefits over the traditional method:

- More reliable results by accounting for 'the 8,8 strat'.
- Gives a certainty value (probability) of the predicted stronghold chunk location, so the player can know whether or not the prediction should be trusted.
- Any number of measurements can be included to increase the precision indefinetly (as opposed to 2 with the traditional triangulation).

2 Model

Let the set of grid points

$$G = \{g_k \in \mathbb{R}^2 : 1408 \leq |g_k| \leq 2688, g_k = (16i+8, 16j+8), i, j \in \mathbb{Z}\}$$

denote the set of all chunk centers (8,8) in the first ring where strongholds can spawn. Let $s \in G$ denote the location of the nearest stronghold to the player (at the time of the first throw). What we ultimately want is to assign a value to the probability $P(s = g_k)$ for each $k = 1, \ldots, K$ based on eye throws, and choose the g_k that maximizes $P(s = g_k)$ as our prediction. At the time of throw n, let γ_{n,g_k} denote the true angle (in degrees) to chunk center g_k from the player, and let α_n denote the angle that is measured by the player. I will assume that the measurement error is normally distributed with mean 0 and variance σ^2, which is pretty standard practice. This gives us:

$$\alpha_n = \gamma_{n,s} + \epsilon_n, \quad \epsilon_n \sim \mathrm{N}(0, \sigma^2).$$

Thus, the conditional probability density of α_n given that $s = g_k$ is

$$p(\alpha_n | s = g_k) = \frac{1}{\sigma\sqrt{2\pi}} e^{-(\alpha_n - \gamma_{n,g_k})^2/2\sigma^2}.$$

Let $\alpha = (\alpha_1, \dots, \alpha_N)$ denote the vector containing all α_n (all throws by the player). If we assume that the measurement errors ϵ_n are independent we get the joint conditional probability density

$$p(\alpha|s = g_k) = \prod_{n=1}^{N} \frac{1}{\sigma\sqrt{2\pi}} e^{-(\alpha_n - \gamma_{n,g_k})^2/2\sigma^2}.$$

Using Bayes' theorem we get the following expression for the probability of chunk location g_k containing the nearest stronghold, given the eye throws:

$$P(s = g_k|\alpha) = \frac{p(\alpha|s = g_k)P(s = g_k)}{p(\alpha)}$$

where $P(s = g_k)$ is the prior distribution of the nearest stronghold, and $p(\alpha)$ is the prior distribution of α. It will be assumed that $P(s = g_k)$ is constant, that is, the nearest stronghold is uniformly distributed on the first ring. This is not true, but a good approximation since in practice only chunks inside a small region will have to be considered (most chunks can be completely disregarded after 2 throws), and inside this small region $P(s = g_k)$ is almost constant. However, one way the model might be improved is by implementing a better calculation of $P(s = g_k)$. Note that the probabilities $P(s = g_k|\alpha)$ have to sum to 1, since the nearest stronghold exists with proability 1 (duh). Now, we can use the fact that $P(s = g_k)/p(\alpha)$ is constant to get

$$\begin{aligned} \sum_{k=1}^{K} P(s = g_k|\alpha) &= 1 \\ \sum_{k=1}^{K} p(\alpha|s = g_k)\frac{P(s = g_k)}{p(\alpha)} &= 1 \\ \implies \frac{P(s = g_k)}{p(\alpha)} &= \frac{1}{\sum_{i=1}^{K} p(\alpha|s = g_i)}, \quad \forall k. \end{aligned}$$

Finally, we obtain the posterior distribution:

$$P(s = g_k|\alpha) = \frac{p(\alpha|s = g_i)}{\sum_{i=1}^{K} p(\alpha|s = g_k)} = \frac{\prod_{n=1}^{N} e^{-(\alpha_n - \gamma_{n,g_k})^2/2\sigma^2}}{\sum_{i=1}^{K}\prod_{n=1}^{N} e^{-(\alpha_n - \gamma_{n,g_k})^2/2\sigma^2}}.$$

The posterior distribution is used to give the player the probability of each chunk containing the stronghold.

3 Notes

In practice the probabilities $p(\alpha|s = g_i)$ are not calculated for all chunks in G, doing so is too computationally intensive, and could not be done 'by hand'. The last two eye throws are used to triangulate the stronghold with the traditional method to get an approximate location, then $p(\alpha|s = g_i)$ is calculated for all chunks in G that are close to the approximated location (7×7 chunks centered around the approximated chunk). Also, it is possible that the assumption that $\mathbb{E}[\epsilon_n] = 0$ is false if the player doesnt know where to aim on the eye, but this can be corrected by a guide, for example. The parameter σ is set depending on how accurate the player is. For players that are just introduced to the tool I have found $\sigma = 0.2$ to be a good value. For an experienced user the value can be set as low as $\sigma = 0.05$. In practice, the smaller σ is, the more 'certain' the algorithm will be that its prediction is correct.

4 Examples

The model has been tested in creative mode, and the model has been better or equally good at predicting the correct chunk than the traditional triangulation method. For a concrete example, I input the 2 throws from this k4yfour video `https://youtu.be/VLefJvyp-nk?t=1651` into my calculator and the correct chunk is predicted with 100% certainty. The throws from the video have been plotted in the following graph: `https://www.desmos.com/calculator/cjqy30rktv` for your convenience. An example of the calculator working in a real run can be seen at `https://youtu.be/zK96gjkLTGc?t=871`.