The Underlying Math Behind Smooth Grayscale Superimposing

This simple explanation will help you understand the mathematics behind this algorithm that mitigates the occurrence of ghost images in superimposed grayscale pictures.

First, the resulting image should produce two rendering results, one overlaying on white background, and the other overlaying on black background. Consider the grayscale of each pixel in the two **rendering results** as $W_{i,j}$ for white background and $K_{i,j}$ with black background; and such in the original input as $w_{i,j}$ and $k_{i,j}$; where the subscript i, j denotes the x and y coordinates correspondingly. The pixels in the actual result image will be denoted as $B_{i,j}$ and $A_{i,j}$ for brightness and alpha accordingly. WLOG, all the variables mentioned above are constrained between [0, 1], with the value of 1 stands for pure white/complete transparent, and the value of 0 stands for pure black/complete opaque.

By simply applying the linear alpha blending algorithm, one may easily derive the following set of equations:

$$\begin{cases} W_{i,j} = A_{i,j}B_{i,j} + \left(1 - A_{i,j}\right) \\ K_{i,j} = A_{i,j}B_{i,j} \end{cases}$$

The unknown variables then can be represented with $W_{i,j}$ and $K_{i,j}$:

$$\begin{cases} A_{i,j} = 1 + K_{i,j} - W_{i,j} \\ B_{i,j} = \dfrac{K_{i,j}}{1 + K_{i,j} - W_{i,j}} \end{cases}$$

Ideally, $W_{i,j}$ should equals to $w_{i,j}$ and so does $K_{i,j}$. However, with the constrain of all variables may only sit between [0, 1], the following inequality must hold.

$$W_{i,j} > K_{i,j} > W_{i,j} - 1$$

In order to comply with such limitation, we may enforce the following inequalities:
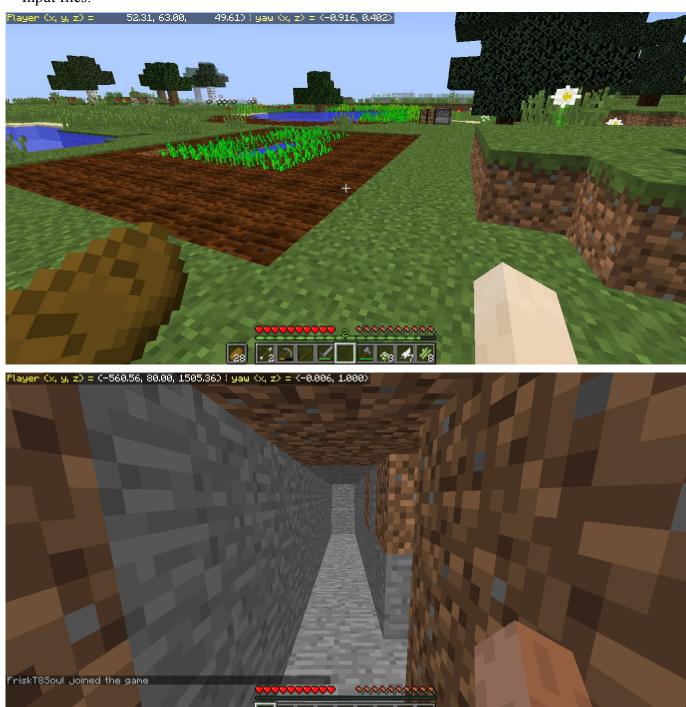
$$\begin{cases} W_{i,j} > 0.5 \\ K_{i,j} \leq 0.5 \end{cases},$$

which may be simply implemented by using:

$$\begin{cases} W_{i,j} = \dfrac{w_{i,j} + 1}{2} \\ k_{i,j} = \dfrac{k_{i,j}}{2} \end{cases}.$$
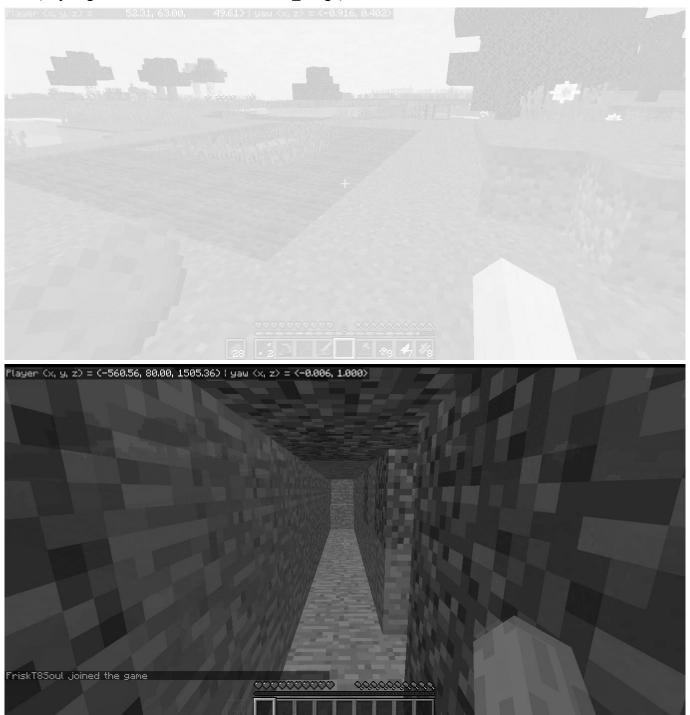
Samples:

Input files:

Player (x, y, z) =   52.31, 63.00,   49.61) | yaw (x, z) = (-0.916, 0.402)

Player (x, y, z) = (-560.56, 80.00, 1505.36) | yaw (x, z) = (-0.006, 1.000)

FriskT8Soul joined the game

Results generated using mentioned algorithm:

Player (x, y, z) =   52.31, 63.00,   49.61> | yaw <x, z> = <-0.916, 0.402>

Player (x, y, z) = (-560.56, 80.00, 1505.36) | yaw <x, z> = <-0.006, 1.000>

FriskT8Soul joined the game

Results generated by using hbl917070's GitHub repository:
(https://github.com/hbl917070/Invisible_image)

With black background, the other image is still visible in hbl917070's implementation, while being completely absence in our algorithm.