

IceMan - Math

Most of these functions correspond to the C math library equivalents, and are called on a per-pixel basis. Given the limited precision of *Fractional* and *Fixed Point* types, some operations must be used with care. (Values are automatically clamped to the legal range).

ice.Image Abs()

Absolute value.

ice.Image Add(*b*)

Add two images.

Parameters

b

second operand (ice.Image).

ice.Image AddWrap(*b*)

Add two images, but wrap the result so that it falls in the range 0-1. Something like a floating point modulo operation.

Parameters

b

second operand (ice.Image).

ice.Image ACos()

Inverse cosine.

ice.Image ASin()

Inverse sine.

ice.Image ATan()

Inverse tangent.

ice.Image ATan2(*x*)

Inverse tangent operation. The image being operated upon is the perpendicular, or y.

Parameters

x

Base (ice.Image).

ice.Image Ceil()

Round up to nearest integer.

ice.Image Cos()

Cosine.

ice.Image Cosh()

Hyperbolic cosine.

ice.Image Divide(*b*)

Divide one image by another: the image being operated upon is the numerator.

Parameters

b

Denominator (ice.Image).

ice.Image Exp()

Inverse natural logarithm.

ice.Image Floor()

Round down to nearest integer.

ice.Image Hypot(*b*)

Compute the hypotenuse of a triangle: the image being operated upon is the first operand.

Parameters

b

Second side of the "triangle" (ice.Image).

ice.Image Ln()

Natural logarithm.

ice.Image Max(*b*)

Result is the maximum of the two inputs.

Parameters

b

Second operand (ice.Image).

ice.Image Min(*b*)

Result is the minimum of the two inputs.

Parameters

b

Second operand (`ice.Image`).

`ice.Image Multiply(b)`

Multiply two images. The multiplier is the image being operated upon.

Parameters

b

Multiplicand (`ice.Image`).

`ice.Image MultiplyAccumulate(b, c)`

Multiply image *a* by *b* and add *c*.

Parameters

b

Scale (`ice.Image`).

c

Offset (`ice.Image`).

`ice.Image MultiplyComplement(b)`

Implements $a * (1 - b)$. *a* is the image being operated upon.

Parameters

b

Second operand (`ice.Image`).

`ice.Image Pow(b)`

Raise this image to the *b* power.

Parameters

b

Second operand (`ice.Image`).

`ice.Image Round()`

Round to nearest integer.

`ice.Image Sign()`

Signum function.

Return value is 1 for positive values, and -1 for negative ones.



Should probably be called `Sgn()`, to conform to common mathematics convention

ice.Image Sine()

Sine function.

ice.Image Sinh()

Hyperbolic sine.

ice.Image SmoothStep(*min*, *max*)

Return 0 if less than the lower threshold, 1 if greater than the upper threshold, and smoothly varying in between. (A cubic is used, with *c1* continuity at both ends).

Parameters

min

Lower threshold (ice.Image).

max

Upper threshold (ice.Image).

ice.Image Subtract(*b*)

Subtract two images. The minuend is the image being operated upon.

Parameters

b

Subtrahend (ice.Image).

ice.Image Tan()

Tangent function.

ice.Image Tanh()

Hyperbolic tangent.