Overview

The asset is composed of an example 2D light and a script. The script offers many properties you can tweak and tune from the inspector. The scope of the script is to simulate a flicking light, by changing intensity, position or colour.

LightFlicker Script

Enabling features

The three features can be enabled/disabled independently and can seamlessly work together. Each of them can be turned on and off through the inspector.

General Functionality

A while loop is performed in a coroutine, one for each feature. The flicking properties are set to random values and each step is delayed by a random waiting time.

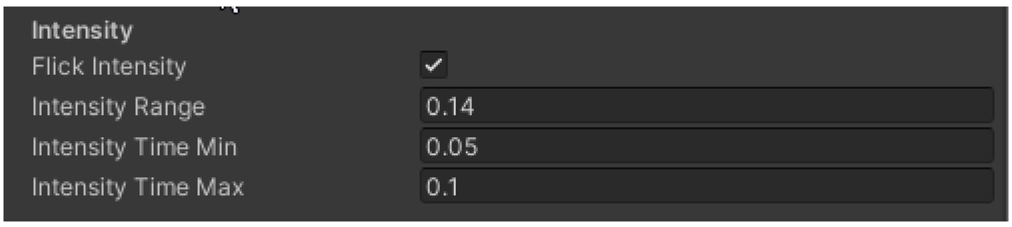
Waiting times

A random waiting time si put before the loop so different instances of flicking lights would start flicking at different times.

The waiting time at the end of a loop step is carried out through the WaitUntil class to ensure the possibility of tweaking the waiting time live.

Intensity Flicker

The intensity flicker takes the default light intensity, that you chose in the light inspector, and changes it in a loop: the new intensity is randomly chosen in a custom range around the default intensity and changed after a random time (also in a custom range).



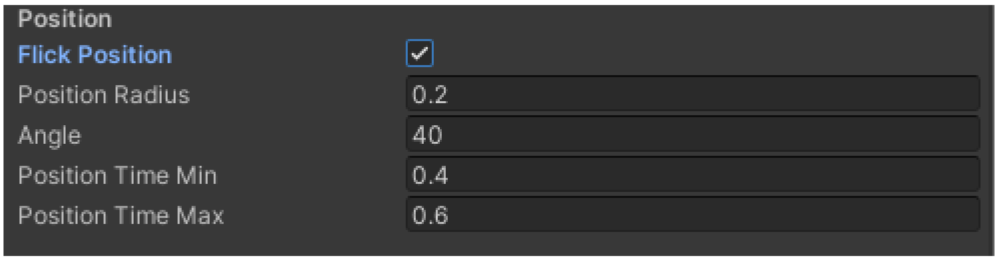
Intensity

Flick Intensity	✓
Intensity Range	0.14
Intensity Time Min	0.05
Intensity Time Max	0.1

This is how the intensity flicker appears in the inspector.

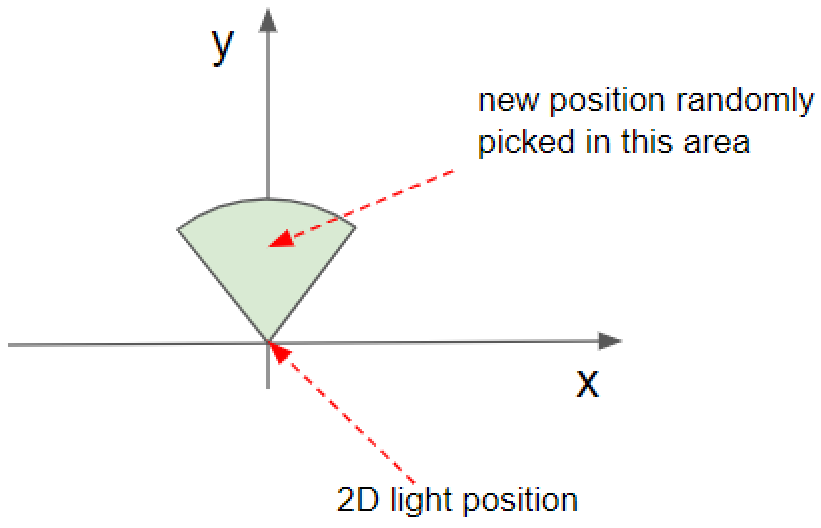
Position Flicker

The position flicker will randomly move the light itself through its transform component. The range of motion is decided by values in the inspector.



Position	
Flick Position	✓
Position Radius	0.2
Angle	40
Position Time Min	0.4
Position Time Max	0.6

Since the idea was to **simulate the flame of a candle** in the breeze, the motion is limited to a circular sector centred in the initial position, and pointing up:



To extract the position randomly, formulas for a random point inside a circle are used and restricted to the circular sector, by using **polar coordinates** with the angular one restricted to an interval around 90°.

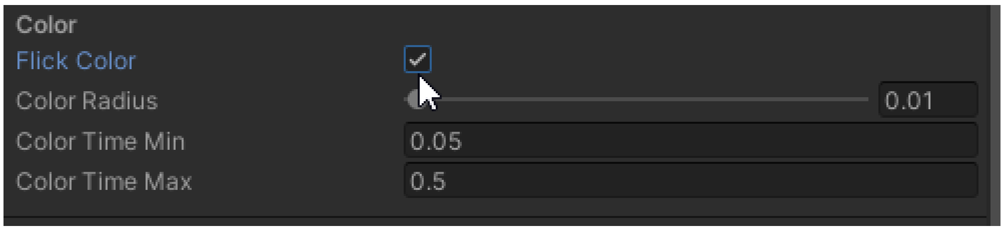
The new position is determined randomly first in polar coordinates, then in cartesian ones as a shift that will be added to the default position.

Small values of the radius are suggested.

Colour Flicker

To reproduce a change in the light colour, the RGB colour space is seen as a **3D space**: the light colour is a point in this space, having coordinates limited in [0, 1] (or [0, 255] if you prefer) along each axis.

A slight shift in the colour is then represented by the extraction of a random point in a **small sphere** centred in the default colour. The sphere radius can be chosen in the inspector, in a range from 0 to 1, but I suggest very small values.


Color

Flick Color	✓
Color Radius	0.01
Color Time Min	0.05
Color Time Max	0.5

The random point extraction uses a property from the random class Random.insideUnitSphere, it is then **scaled** to the chosen radius.

Very small values of radius are suggested to avoid the *disco* effect.