# Movit: The modern video toolkit

A high-quality, high-performance, open-source* library for video filters

* GPLv2+

# High-performance?

The road from the 80s to 2014 and back

# ANSI C (1989)

```
int i;
for (i = 0; i < width * height; ++i) {
        out[i] = a[i] * (1.0 - f) + b[i] * f;
}
```

# Save one multiply

```
int i;
for (i = 0; i < width * height; ++i) {
        out[i] = a[i] + (b[i] - a[i]) * f;
}
```

# 16.16 fixed-point (typical 90s code)

```
int fi = lrintf(f * 65536.0);
const int ROUND = 32768;
int i;
for (i = 0; i < width * height; ++i) {
        out[i] = a[i] + (((b[i] - a[i]) * fi + ROUND) >> 16);
}
```

# MMX (1997)

```
const int fi = lrintf(f * 256.0), finv = 256 - fi, ROUND = 128;
int i = 0;
#ifdef __MMX__
const __m64 zero = _mm_setzero_si64(), roundvec = _mm_set1_pi16(ROUND);
const __m64 fvec = _mm_set1_pi16(fi), finvvec = _mm_set1_pi16(finv);
const __m64 *avec = (__m64 *)a, *bvec = (__m64 *)b;
__m64 *outvec = (__m64 *)out;
for ( ; i < width * height; i += 8) {
        __m64 aval = avec[i / 8], bval = bvec[i / 8];
        __m64 a_lo = _mm_unpacklo_pi8(aval, zero), a_hi = _mm_unpackhi_pi8(aval, zero);
        __m64 b_lo = _mm_unpacklo_pi8(bval, zero), b_hi = _mm_unpackhi_pi8(bval, zero);
        __m64 out_lo = _mm_add_pi16(_mm_mullo_pi16(finvvec, a_lo), _mm_mullo_pi16(fvec, b_lo));
        __m64 out_hi = _mm_add_pi16(_mm_mullo_pi16(finvvec, a_hi), _mm_mullo_pi16(fvec, b_hi));
        out_lo = _mm_srli_pi16(_mm_add_pi16(out_lo, roundvec), 8);
        out_hi = _mm_srli_pi16(_mm_add_pi16(out_hi, roundvec), 8);
        outvec[i / 8] = _m_packuswb(out_lo, out_hi);
}
#endif
for ( ; i < width * height; ++i) {
        out[i] = (a[i] * fi + b[i] * finv + ROUND) >> 8;
}
```

# Multicore (early 2000s)

```
// All the stuff from the previous example, plus support for SSE2, AVX2, etc...

pthread_t threads[NUM_CORES];
for (int i = 0; i < NUM_CORES; ++i) {  // C99 has happened in the meantime!
	pthread_create(&threads[i], NULL, process_slice, (void *)i);
}
// Wait for all the cores to come back, hope that all of them were idle...
for (int i = 0; i < NUM_CORES; ++i) {
	pthread_join(threads[i], NULL);
}
```

# Frei0r blend plugin (2006)

```
const uint8_t bf = (const uint8_t) (255 * blend_factor);
const uint8_t one_minus_bf = (255 - bf);
uint32_t w = size;
uint32_t b;

while (w--)
  {
    for (b = 0; b < NBYTES; b++)
      dst[b] = (src1[b] * one_minus_bf + src2[b] * bf) / 255;

    src1 += NBYTES;
    src2 += NBYTES;
    dst += NBYTES;
  }
```

# 2014: GPUs are no longer uncommon


Tim "Avatar" Bartel, CC-BY-2.0

# 2014: GPUs are no longer uncommon

flickr.com/photos/footfun, CC-BY-2.0

# 2014: GPUs are no longer uncommon

Mia Aasbakken, used with permission

# 2014: GPUs are no longer uncommon


Pantelotteriet Panto, Apeland Informasjon/Katrine Lunke

# GLSL (2004)

```
vec2 tc = gl_MultiTexCoord0.xy;
vec4 first = texture2D(texA, tc);
vec4 second = texture2D(texB, tc);
gl_FragColor = first + vec4(f) * (second - first);
```

# Adjusted for inclusion in Movit

```
vec4 FUNCNAME(vec2 tc) {
	vec4 first = INPUT1(tc);
	vec4 second = INPUT2(tc);
	return first + vec4(PREFIX(f)) * (second - first);
}
```

# High-quality?

Or: There is more than one way to mess it up

# Fade done in gamma space

Linear light (correct)

sRGB (incorrect)

# Rec. 2020 misinterpreted as sRGB (Rec. 709)

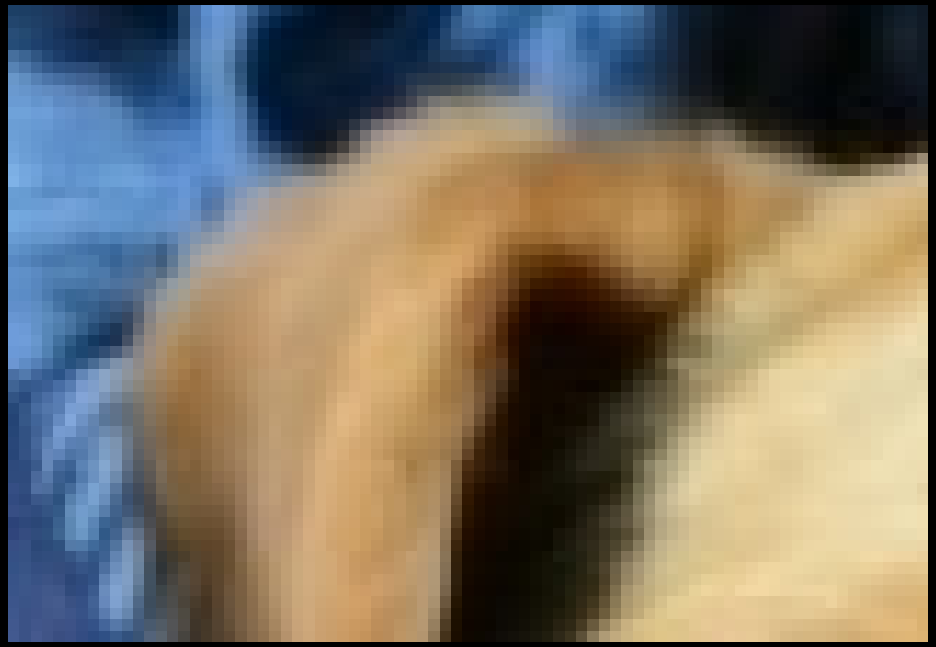
(emphasized; done three times)

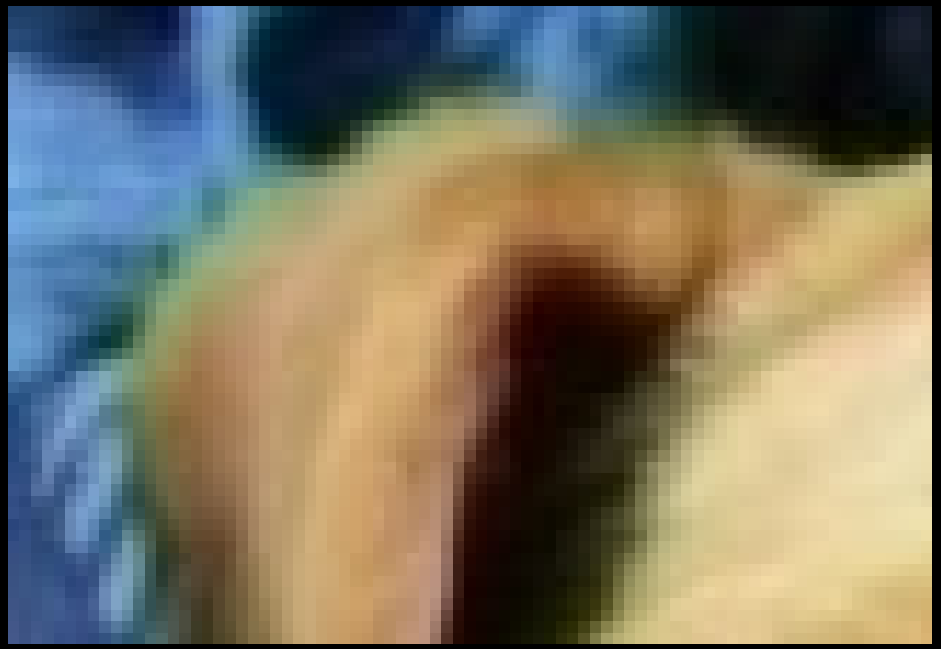
# Wrong chroma subpixel placement

(YCbCr 4:1:0)

# Wrong chroma subpixel placement

(YCbCr 4:1:0)

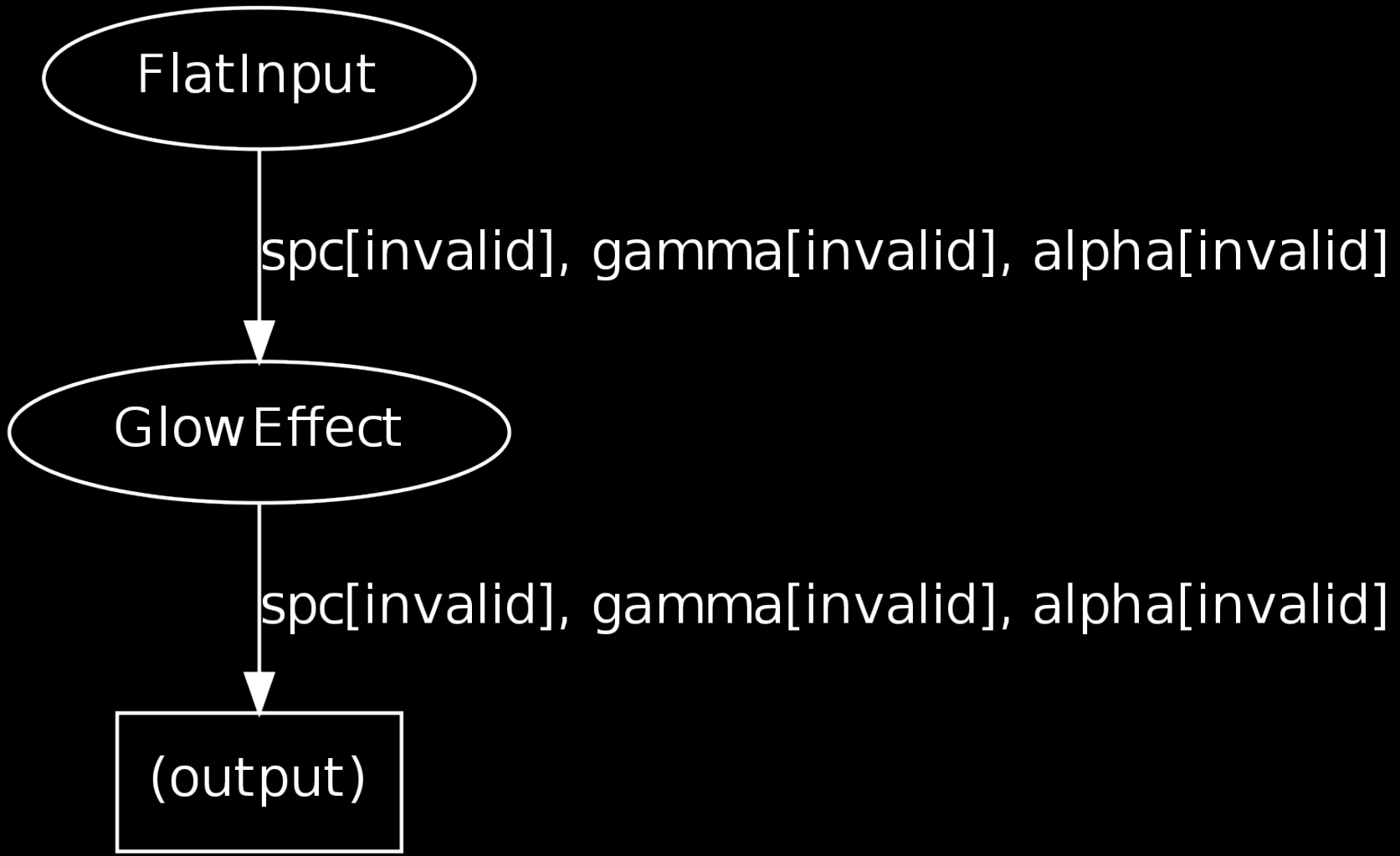


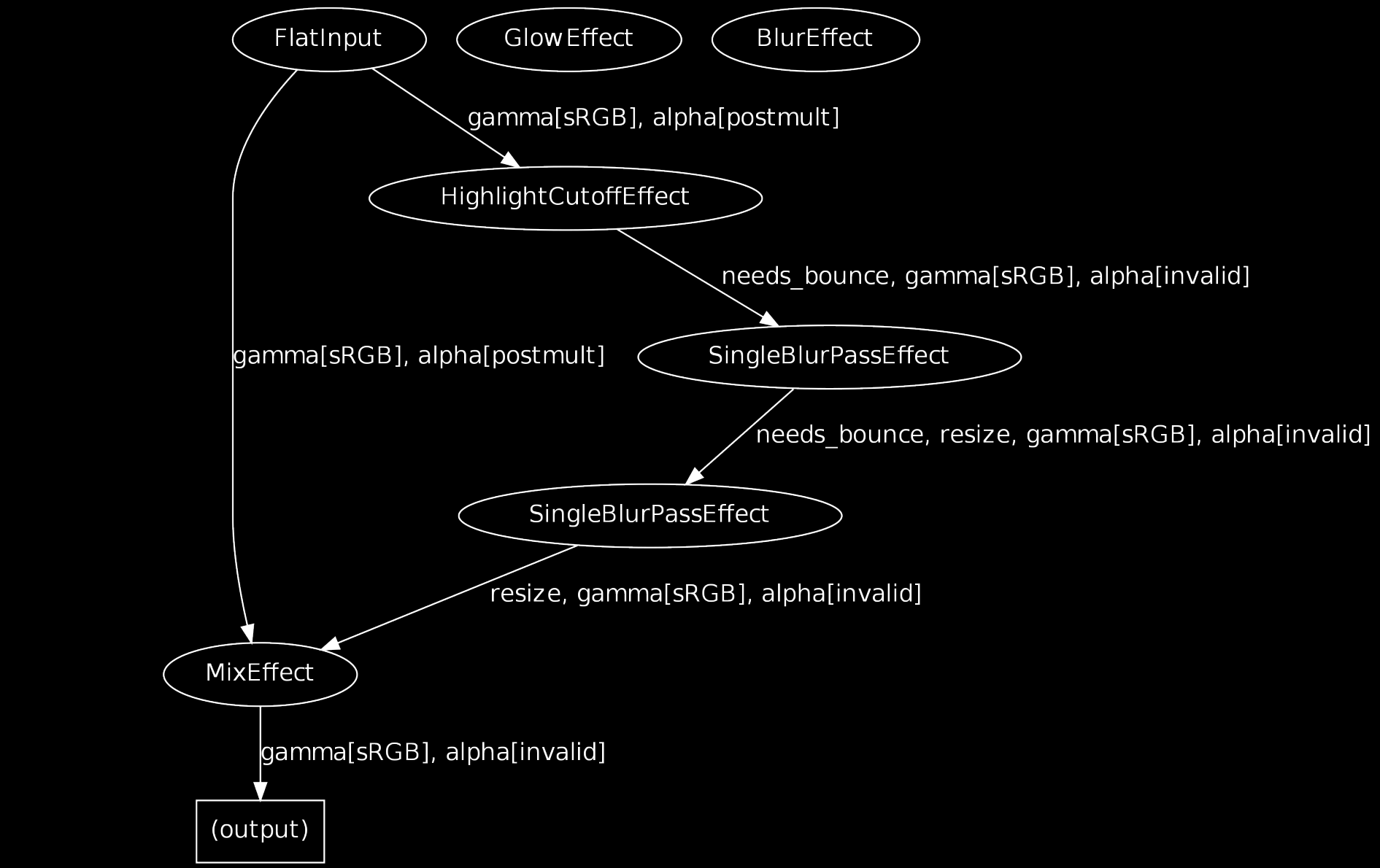


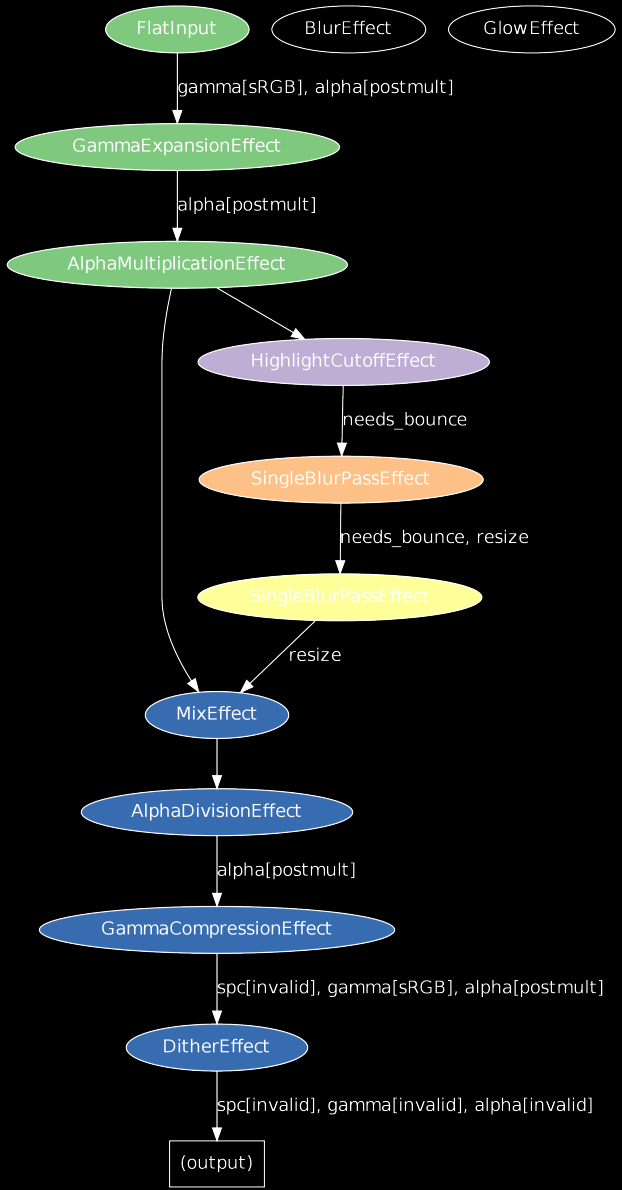
```
$ wc -l *.cpp | grep total
  10625 total

$ wc -l *test*.cpp | grep total
  4852 total
```

# My first Movit chain

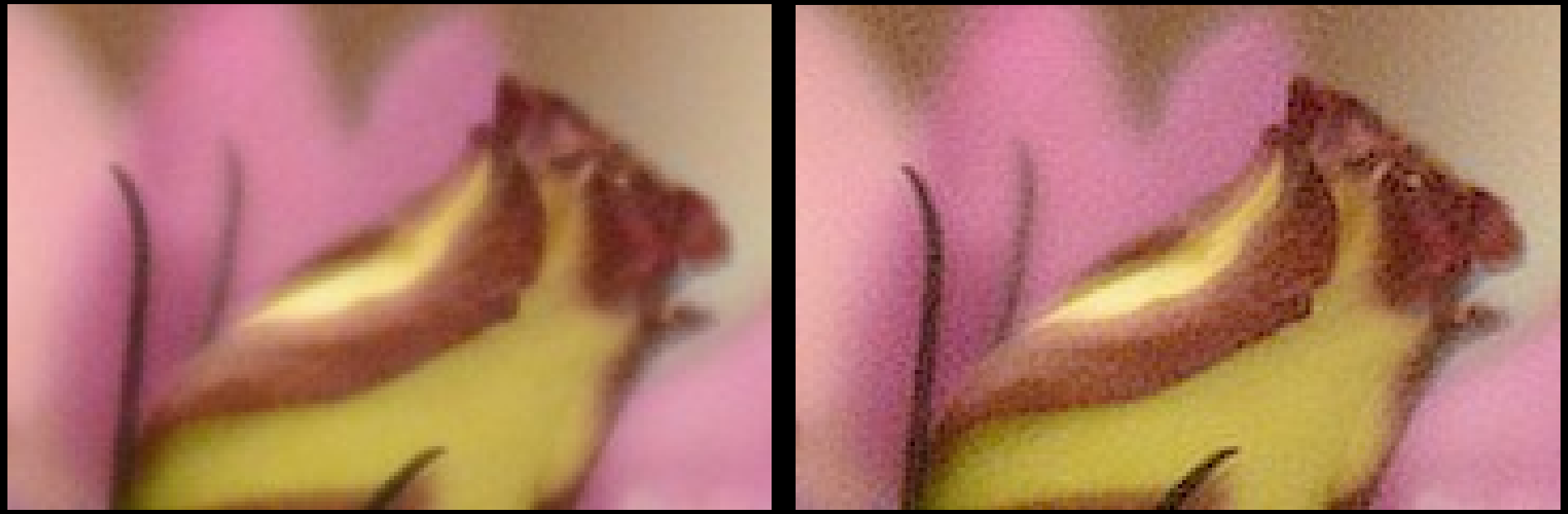
```
ImageFormat inout_format(COLORSPACE_sRGB, GAMMA_sRGB);
FlatInput *input = new FlatInput(inout_format,
        FORMAT_BGRA_POSTMULTIPLIED_ALPHA,
        GL_UNSIGNED_BYTE, width, height);
chain.add_input(input);
chain.add_effect(new GlowEffect());
chain.add_output(inout_format,
        OUTPUT_ALPHA_FORMAT_POSTMULTIPLIED);
chain.set_dither_bits(8);
chain.finalize();
// ....
input->set_pixel_data(buf);
chain.render_to_screen();
```


FlatInput
spc[invalid], gamma[invalid], alpha[invalid]
GlowEffect
spc[invalid], gamma[invalid], alpha[invalid]
(output)


FlatInput
GlowEffect
BlurEffect
gamma[sRGB], alpha[postmult]
HighlightCutoffEffect
needs_bounce, gamma[sRGB], alpha[invalid]
gamma[sRGB], alpha[postmult]
SingleBlurPassEffect
needs_bounce, resize, gamma[sRGB], alpha[invalid]
SingleBlurPassEffect
resize, gamma[sRGB], alpha[invalid]
MixEffect
gamma[sRGB], alpha[invalid]
(output)

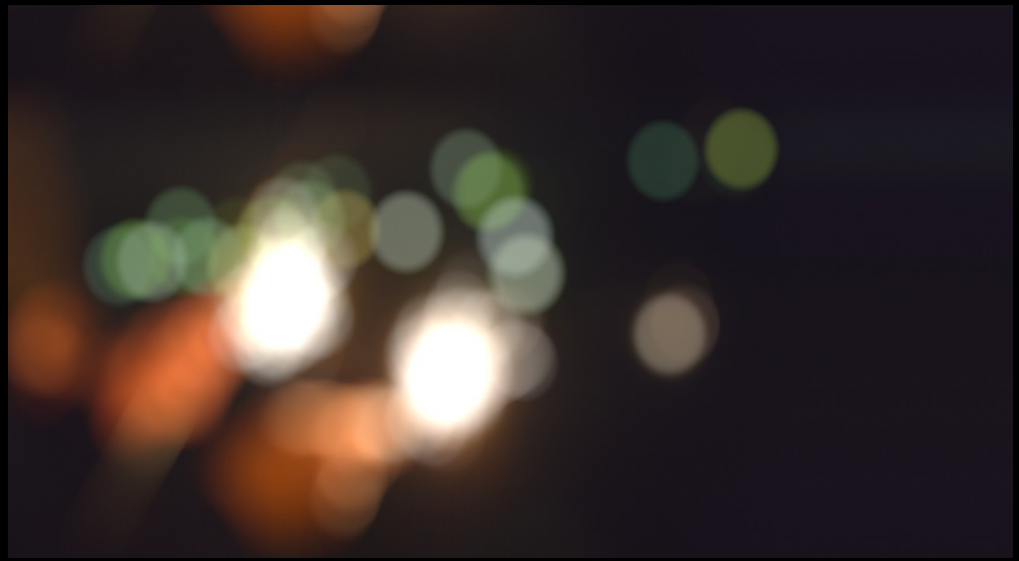

FlatInput
BlurEffect
GlowEffect
gamma[sRGB], alpha[postmult]
GammaExpansionEffect
alpha[postmult]
AlphaMultiplicationEffect
HighlightCutoffEffect
needs_bounce
SingleBlurPassEffect
needs_bounce, resize
SingleBlurPassEffect
resize
MixEffect
AlphaDivisionEffect
alpha[postmult]
GammaCompressionEffect
spc[invalid], gamma[sRGB], alpha[postmult]
DitherEffect
spc[invalid], gamma[invalid], alpha[invalid]
(output)

# More Movit effects




ROPPONGI ROPPONGI
Hennessy


ROPPONGI ROPPONGI
Hennessy



(Work in progress, coming soon)

(Work in progress, coming soon)

# Demo time!

# Future work

- More filters
- Better YCbCr handling
- Handling of interlaced content
- Whatever clients need (within reason…)

# Thank you!

http://movit.sesse.net