A Low-Distortion Map Between Triangle and Square

Eric Heitz

A Low-Distortion Map Between Triangle and Square

Eric Heitz

Unity Technologies

Abstract

We introduce a low-distortion map between triangle and square. This mapping yields an area-preserving parameterization that can be used for sampling random points with a uniform density in arbitrary triangles. This parameterization presents two advantages compared to the square-root parameterization typically used for triangle sampling. First, it has lower distortions and better preserves the blue-noise properties of the input samples. Second, its computation relies only on arithmetic operations (+, *), which makes it faster to evaluate.

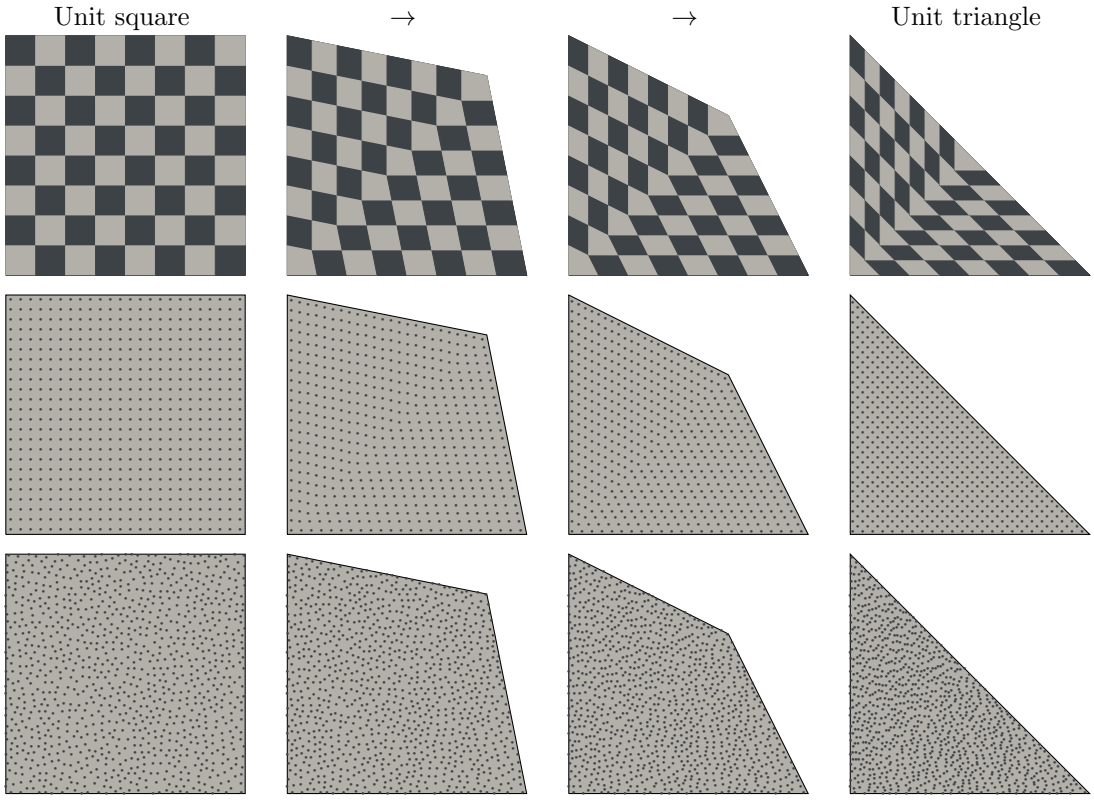

Figure 1: **A low-distortion map between triangle and square.** We morph the unit square by compressing it along the diagonal such that it becomes a right-angle triangle.

1 Evaluation of the parameterization

Problem statement Given an arbitrary triangle (V_1, V_2, V_3), we want to pick a random point in the triangle with uniform probability density. The randomization is given by two uniform random numbers s_1 and s_2 that can be seen as the coordinates of a random point in the unit square. In order to preserve the properties of these unit-square samples (such as their stratification or blue-noise distribution), we are looking for an *area-preserving parameterization*, i.e. a *bijection* between the unit square and the triangle of *constant Jacobian*, and that has *low distortions*.

1. Mapping the unit square to the unit triangle Our idea is in the same spirit as Shirley's concentric mapping [SC97]: our parameterization is based on a morphing that maps the unit square to the unit triangle without introducing too many distortions. This is shown in Fig 1. To compute this mapping, we proceed as in Fig 2: for a given input point (u_1, u_2) in the unit square, we compute its distance l to the edge of the square along the diagonal and we divide this distance by 2 to obtain a point (t_1, t_2) in the unit triangle. Hence, this parameterization maps diagonal-aligned segments of the unit square to their first half. This is why the parameterization has a constant Jacobian that equals 2 everywhere, i.e. it is area-preserving.

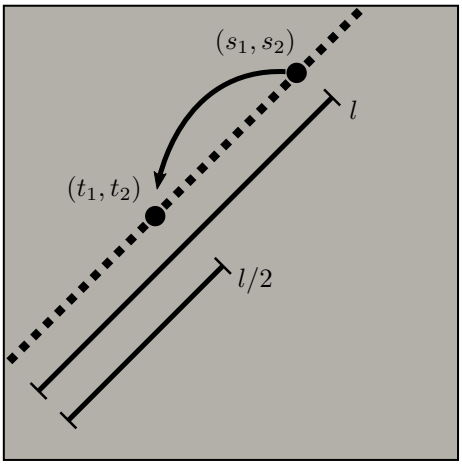

Figure 2: **Evaluation of the parameterization.**

In practice, we map the point by dividing its coordinates by 2 and adding the offset due to the projection on the edge, as shown in Listing 1.

Listing 1: Evaluation of the parameterization.

```
void square2triangle(
    in float s1, in float s2, // input: unit-square point
    out float t1, out float t2 // output: unit-triangle point
)
{
    t1 = 0.5f * s1;
    t2 = 0.5f * s2;
    float offset = t2 - t1;
    if(offset>0)
        t2 += offset;
    else
        t1 -= offset;
}
```

2. Mapping to an arbitrary triangle Finally, we remap the coordinates (t_1, t_2) inside the unit triangle to an arbitrary triangle (V_1, V_2, V_3):

$$V = t_1 V_1 + t_2 V_2 + (1 - t_1 - t_2) V_3. \tag{1}$$

2 Comparison against the square-root parameterization

To our knowledge, the most widely used area-preserving parameterization for sampling triangles is the *square-root parameterization* [Tur90]:

$$V = (1 - \sqrt{s_1}) V_1 + \sqrt{s_1} \, (s_2 V_2 + (1 - s_2) V_3) \, . \tag{2}$$

Performance In our CPU bench tests, our parameterization is about 20% faster to evaluate than the square-root parameterization.

Quality of the parameterization In Fig. 3 and 4, we compare the distortions of the parameterizations. The square-root parameterization has large distortions near the frst vertex and the properties of the input samples are lost in this region. Our parameterization procudes a more uniform distribution of the point samples.

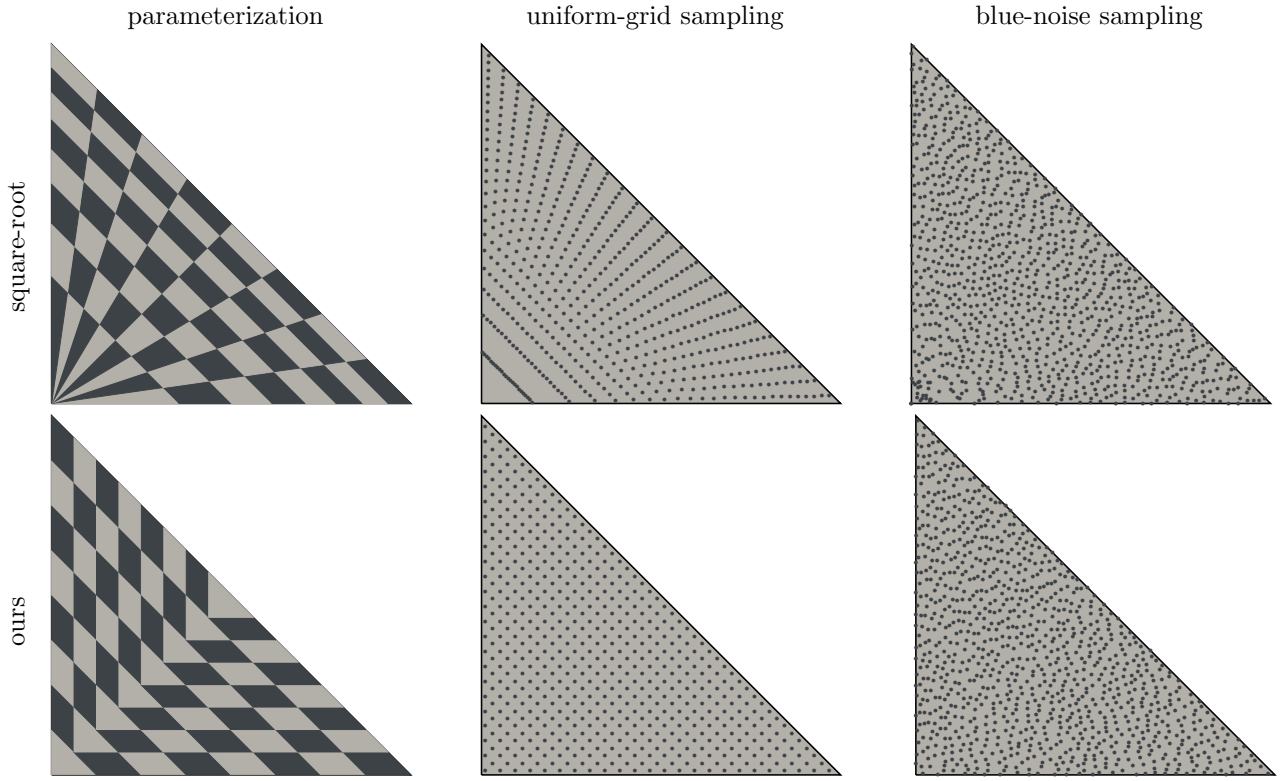

Figure 3: **Comparison on the unit triangle.**

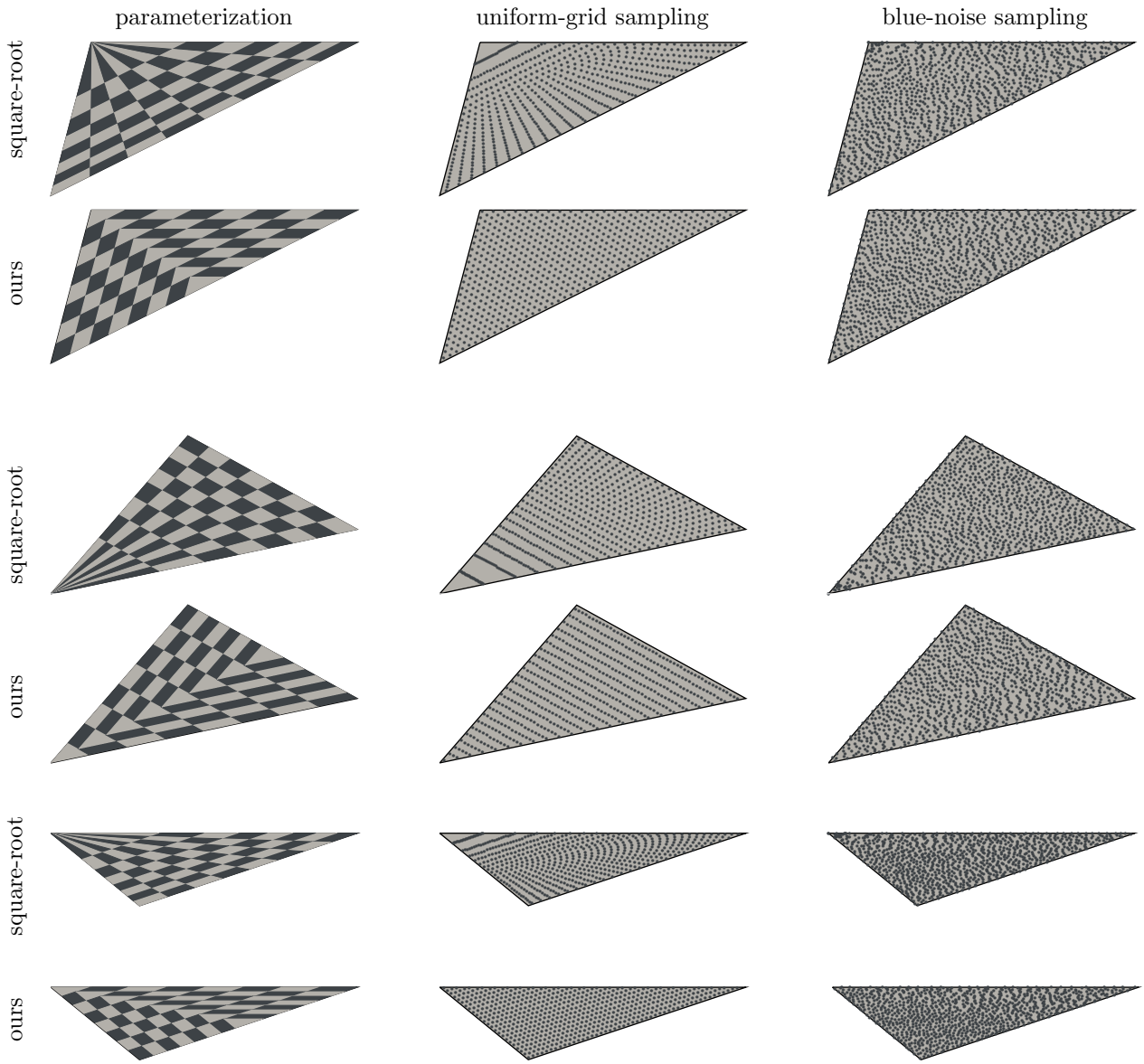

Figure 4: **Comparisons on arbitrary triangle.**

References

[SC97] Peter Shirley and Kenneth Chiu. A low distortion map between disk and square. *J. Graph. Tools*, 2(3):45–52, December 1997.

[Tur90] Greg Turk. Generating random points in triangles. In Andrew S. Glassner, editor, *Graphics Gems*, pages 24–28. Academic Press, 1990.

A Optimized implementation

After the first version of this technical report was released, Andrew Kensler proposed to reorganize the calculations in order to save some instructions. This optimized implementation is also slightly more accurate in single precision compared to a double-precision reference.

Listing 2: Evaluation of the parameterization (optimized implementation).

```
// maps a unit-square point (x, y) to a unit-triangle point
void square2triangle(in out float x, in out float y)
{
    if ( y > x ) {
        x *= 0.5f;
        y -= x;
    } else {
        y *= 0.5f;
        x -= y;
    }
}
```