Non-Linearly Quantized Moment Shadow Maps

Christoph Peters

2017-07-30

High-Performance Graphics 2017

These slides include presenter's notes for your convenience.

1

In this presentation we discuss non-linearly quantized moment shadow maps. The notes you are reading are a best effort to make the slides stand for themselves. Please note that some of the slides use animations that will only be visible when the presentation is viewed as slide show in PowerPoint.

In a nutshell

- Fast antialiased hard shadows,
- Minimal light leaking,
- 32 or 64 bits per shadow map texel,
- Cost matches variance shadow mapping.

2

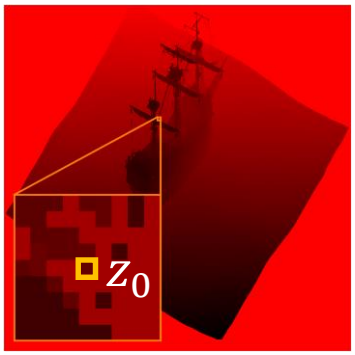
Our novel technique offers antialiased hard shadows. That is valuable because shadow map aliasing is a widely seen artifact in modern games. Of course the technique also has to be fast to find use. CLICK And it has to be robust which means having little light leaking artifacts in this case. CLICK There are variants using either 32 or 64 bits per shadow map texel. In both cases, the run time cost is comparable to that of variance shadow mapping.

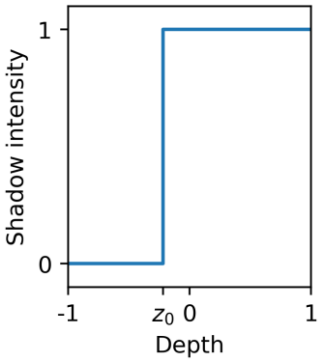
Moment shadow mapping

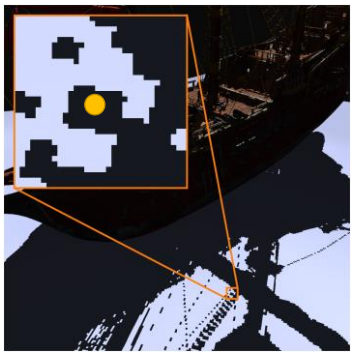
A recap

The technique extends moment shadow mapping so I will start with a recap of that.

Shadow mapping [Williams78]







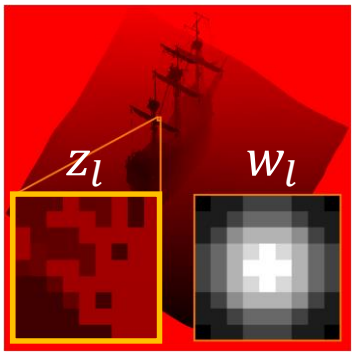
Shadow map Depth distribution Scene

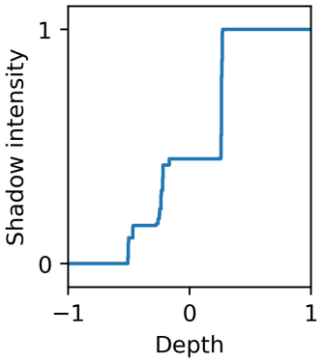
$$Z = \delta_{z_0}$$

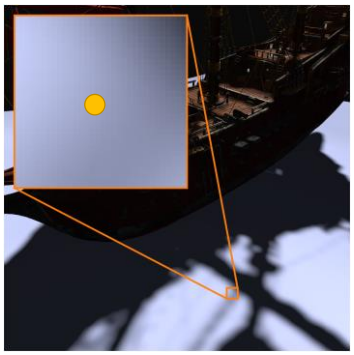
4

In real-time rendering our shadows are usually based on shadow maps. We render a view of the scene from the point of view of the light source and store depth values. Looking at a single texel in the shadow map tells us that for the corresponding light ray we have no shadow up to some depth z_0 and full shadow beyond this point. This is a simple technique but it takes samples in light and screen space without any filtering so it suffers from unacceptable aliasing.

Percentage-closer filtering [Reeves87]





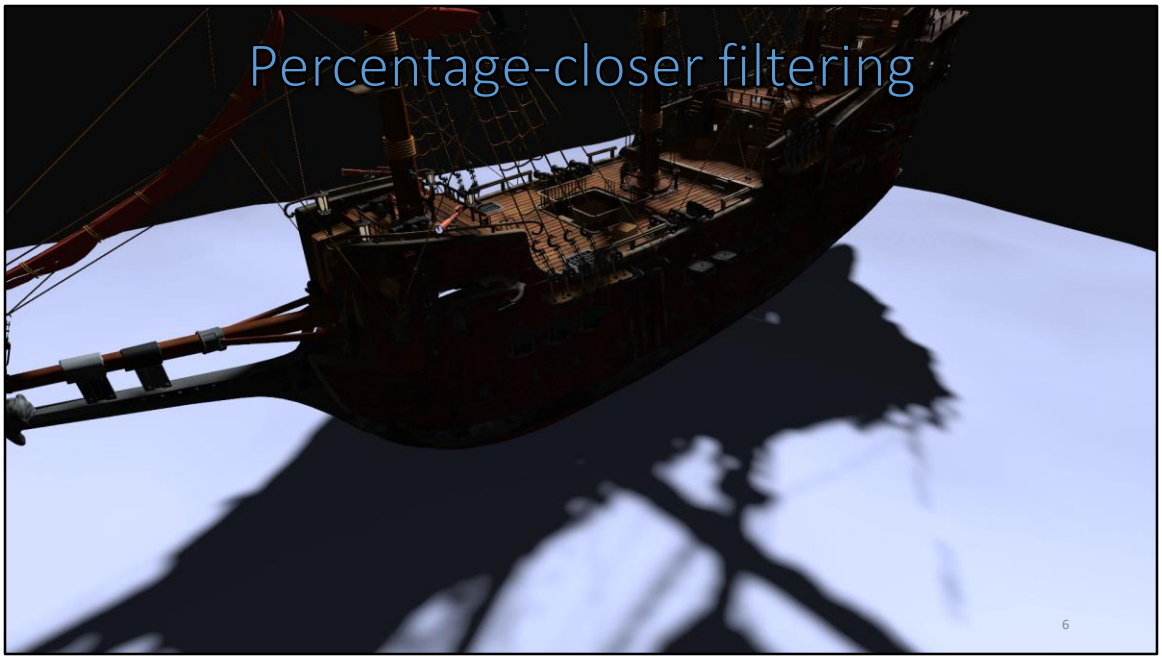


Shadow map Depth distribution Scene

$$Z = \sum_{l=0}^{n-1} w_l \cdot \delta_{z_l}$$

5

The most widely used technique to diminish shadow map aliasing is percentage-closer filtering. Here we do not only consider a single shadow map sample but a larger filter region. The results for every single sample are combined using weights from a filter kernel such as a Gaussian. This leads to a filtered shadow that looks more plausible.



Percentage-closer filtering is a useful technique but it is either too costly or does not diminish aliasing sufficiently. In this video example, we use 100 samples per pixel and still observe strong aliasing.

Moment shadow maps [Peters15]

- Store z, z^2, z^3, z^4 in RGBA,
- Filtering generates 4 moments:

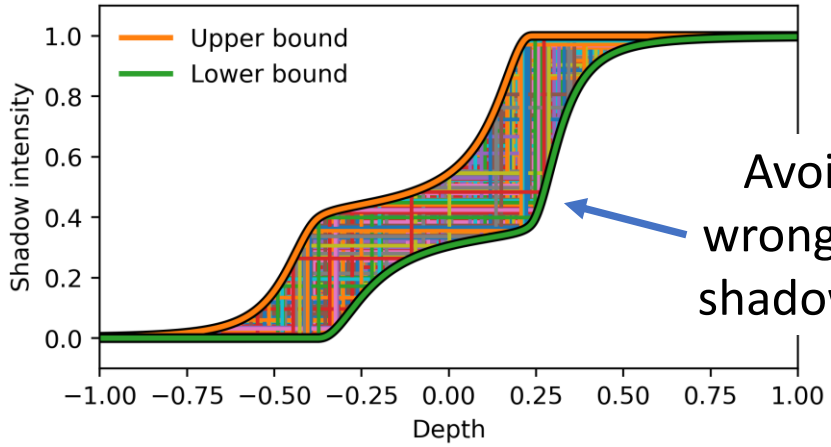
$$b = \begin{pmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \end{pmatrix} = \sum_{l=0}^{n-1} w_l \cdot \begin{pmatrix} z_l \\ z_l^2 \\ z_l^3 \\ z_l^4 \end{pmatrix} \in \mathbb{R}^4$$

7

To overcome these problems, a moment shadow map stores some additional information. It has four channels storing four powers of the light space depth value. Unlike a common shadow map, it is useful to apply a filter such as a separable blur directly to the moment shadow map. Doing so leads to linear combinations of the vectors stored in the texels. The resulting vectors are called moment vectors. Their entries are four moments of the depth distribution.

Optimal lower bound

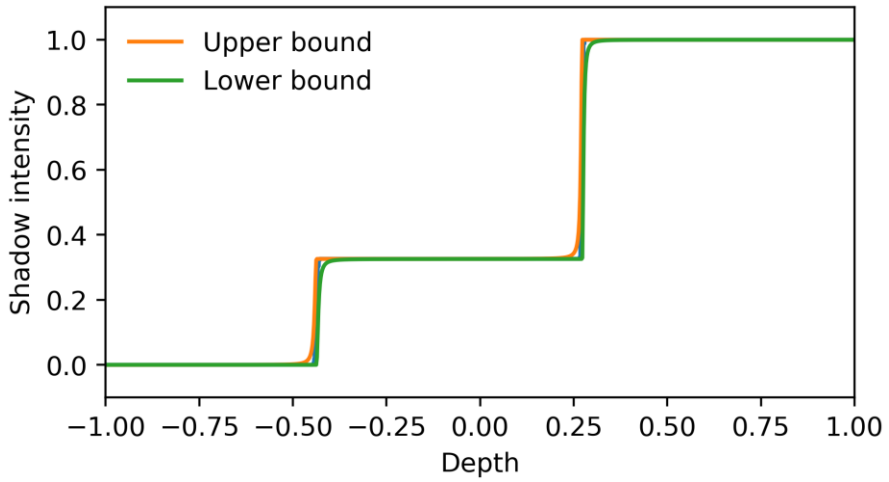
$b_1 = 0, b_2 = 0.093, b_3 = -0.013, b_4 = 0.013$



8

This additional information enables a heuristic reconstruction. Here is an example how that works. We have a ground truth depth distribution but we do not actually know it. The moment shadow map only provides the four moments at the top. Of course the ground truth is not uniquely characterized by these moments. CLICK Here is an example of a different depth distribution that yields the same moments. CLICK And here is another one. CLICK This one also has the same moments. CLICK And here are a few more examples. It is apparent that there is an infinite-dimensional space of possible reconstructions. CLICK What is important is that all of them lie between specific lower and upper bounds. Moment shadow mapping offers highly optimized algorithms to compute these bounds. CLICK For shadows we want to use the lower bound because this way we avoid that surface patches shadow themselves.

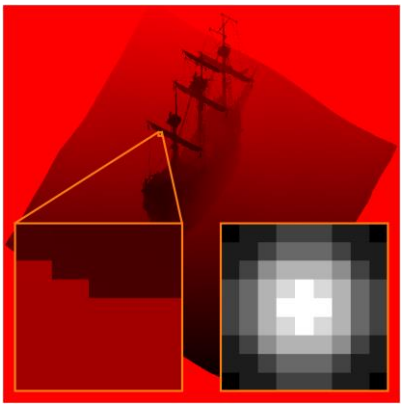
Bounds are often very narrow



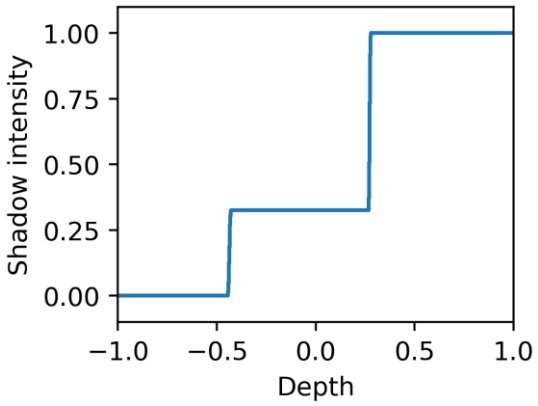
9

Looking at the previous example, you may think that this reconstruction is not very accurate because the bounds are quite far apart. However, the results are much better in a far more relevant case. In this example, the ground truth essentially only uses two different depth values. CLICK For such a ground truth the bounds are extremely tight.

Bounds are often very narrow



Shadow map

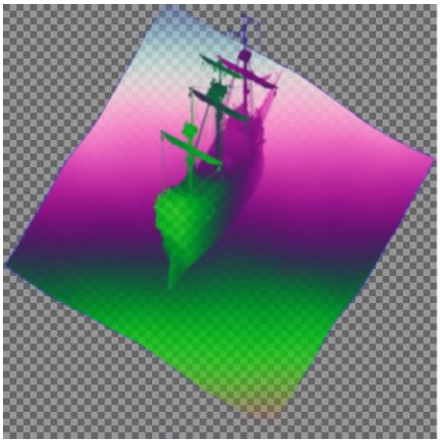


Depth distribution

10

This case is very important because it corresponds to a situation that occurs commonly in shadow map filtering. The filter region only contains the silhouette of an object. Thus, there are only two surfaces above and below this silhouette and the depth varies only minimally within these surfaces. For such a case, moment shadow mapping offers a nearly perfect reconstruction.

Moment shadow mapping with 8x MSAA



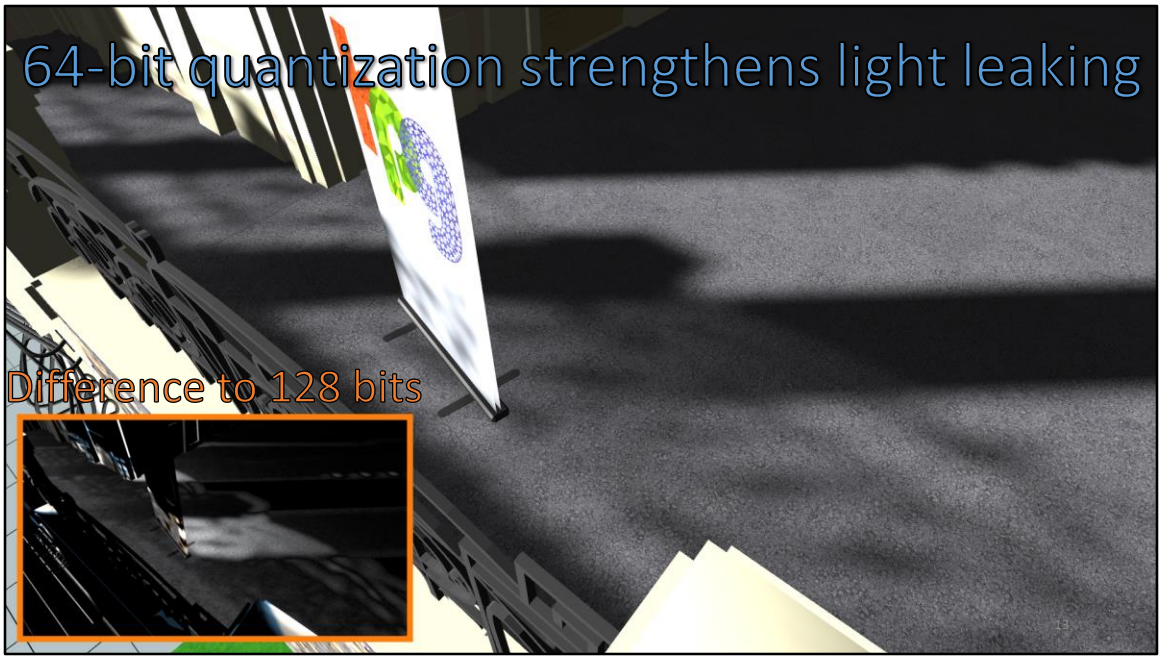
Moment shadow map

Scene

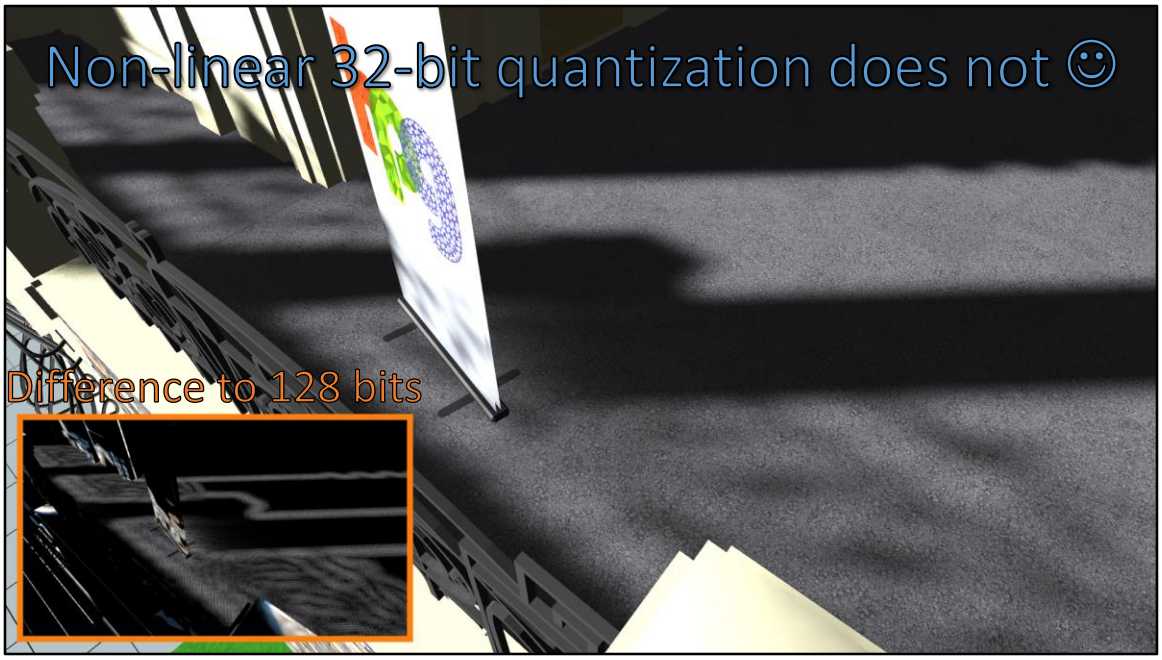
11

Now here you can see moment shadow mapping in action. To the left there is the filtered moment shadow map with its four channels. The checkerboard serves to visualize the alpha channel. To the right you can see the shaded scene. As you can see, there is much less aliasing than with percentage-closer filtering. This is owed to the fact that we use 8x multisample antialiasing to generate the moment shadow map. This works naturally with moment shadow maps but would be very expensive with percentage-closer filtering. We also use a 9x9 separable Gaussian blur here.

Here is another result of moment shadow mapping. The quality is very high. Even short range shadows in places where shadow casters meet are reproduced accurately. However, we have used 128 bits per texel of the moment shadow map storing each moment in a single-precision float. This is quite expensive.



There is another alternative. When an appropriate linear transform is applied before storage, it is viable to use only 64 bits per texel of the moment shadow map. However, the quantization introduces rounding errors. To avoid that these errors break the reconstruction, a bias has to be applied. Unfortunately, this bias introduces some light leaking in short-range shadows. The technique is still useful and has been used in production as is but there is room for improvement.



With our non-linear quantization, we get no increase in light leaking even at 32 bits per texel.

Outline

- Non-linear quantization,
- Fast filtering,
- Bilinear interpolation.

15

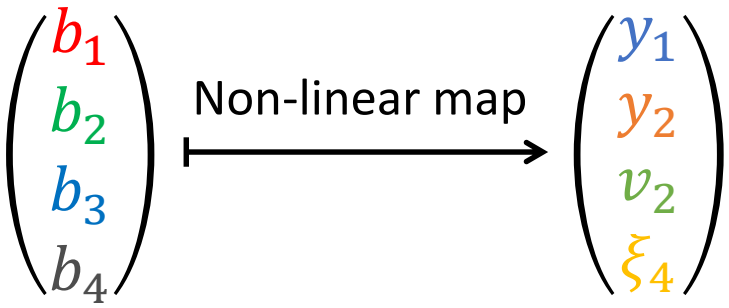
To get there, we will first discuss the quantization scheme itself. Then we find that it is not a good match for the way a moment shadow map is commonly filtered. Thus, we provide a more sophisticated scheme that offers a considerable speedup. Finally, we figure out how to do without hardware-accelerated bilinear interpolation.

Non-linearly quantized moment shadow maps

16

So lets start with the quantization scheme.

A non-linear representation of moments



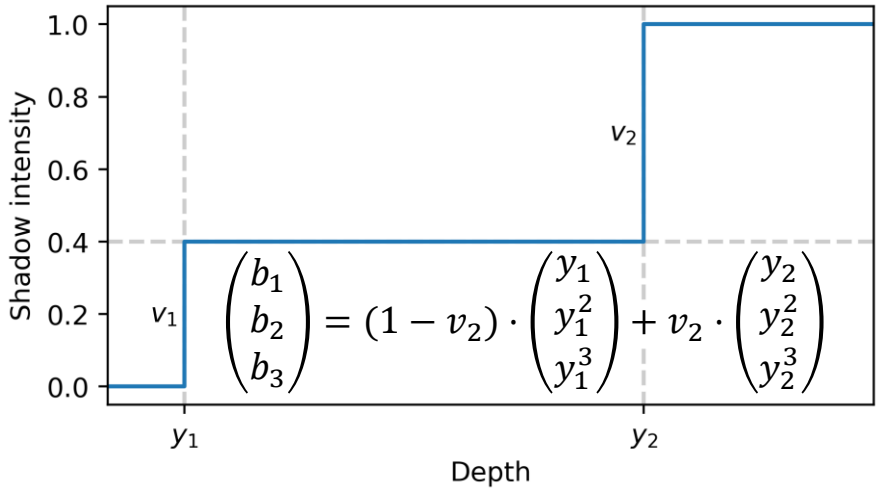
Vulnerable to rounding errors

Robust to rounding errors

17

We start from a vector of filtered moments. This vector is vulnerable to rounding errors such that we cannot store it as compactly as we would like. We seek a different representation through four scalars that is more robust to rounding errors and can be quantized more aggressively. Since the original moment shadow mapping already uses an optimized linear transform, we can only hope for an improvement with a non-linear map. On the next slides I will explain the four quantities to the right.

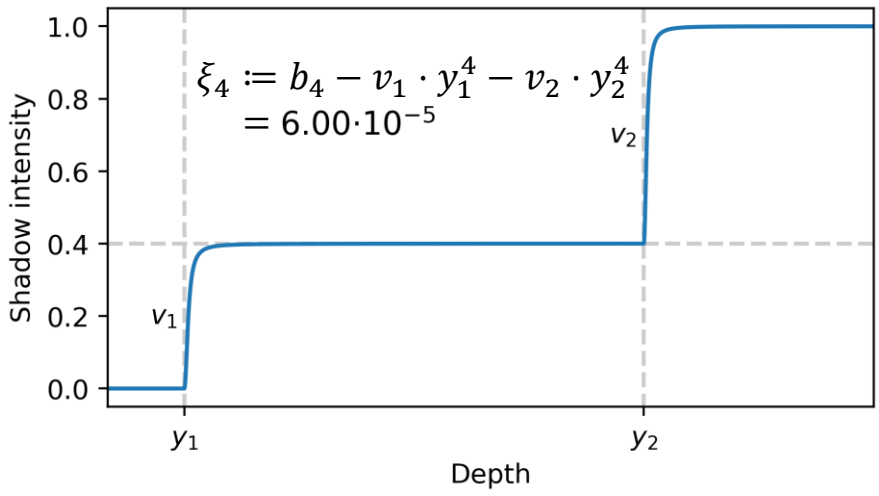
3 moments as 2 depths and 1 weight



18

Lets reconsider the case where moment shadow mapping yields a perfect reconstruction. The entire depth distribution is characterized by the two depth values y_1 and y_2 and two weights v_1 and v_2 which add up to one. CLICK It is easy to compute the first three moments of this depth distribution. Prior work also provides us with an efficient algorithm to go the other way. Thus, we can represent the first three moments through two depths and one weight. We know how to store depth values and normalized weights and we understand how errors change the depth distribution. Therefore, these three quantities provide a useful non-linear representation.

The offset of the 4th moment

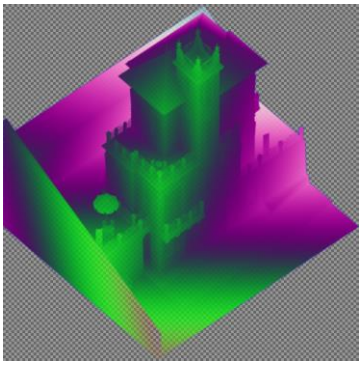


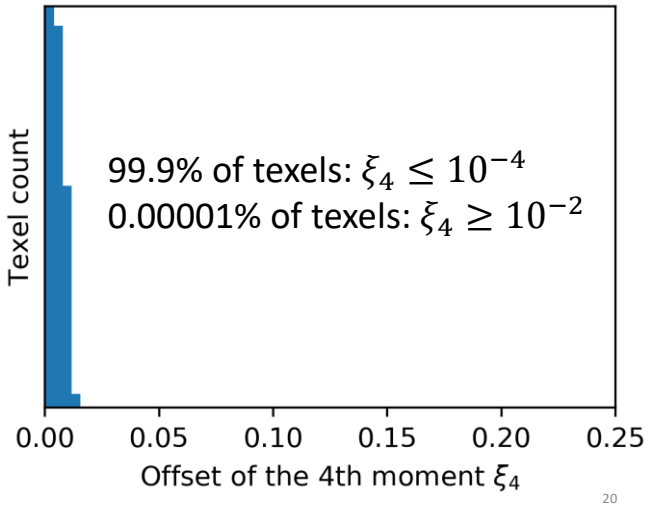
19

Of course we must not drop the fourth moment entirely. Instead of storing it directly, we store its difference to the fourth moment that we would have if the perfect reconstruction case were present. If this is indeed the case, this offset of the fourth moment is zero. Now lets see what happens for greater values. CLICK We increase the value slowly. As we do so, the lower bound that serves as reconstruction is smoothed out a little bit. At a value of 6*10^-5 it already differs from the perfect reconstruction considerably. When we use linear 64-bit quantization, errors introduced into the offset of the fourth moment are in this magnitude. This slightly smoothed out reconstruction is the reason for the light leaking in short-range shadows. To avoid that, it is important to preserve small values accurately. CLICK This is not trivial because the offset of the fourth moment can also take much larger values when there are more surfaces in the filter region.

The offset of the 4th moment is small

Moment shadow map





20

To understand what we are dealing with, we should take a look at the distribution of this offset. To the left you can see a filtered moment shadow map. To the right, there is a histogram showing values of the offset of the fourth moment on the x-axis and the corresponding number of texels on the y-axis. Obviously, this histogram is very non-uniform. 99.9% of all texels have a tiny value below 10^-4. Only very few texels have moderately large values. Essentially, this is good news because it means that the case with perfect reconstruction is extremely common. On the other hand, it poses a problem for storing the offset because even tiny rounding errors may be intolerable.

Warping the offset of the 4th moment

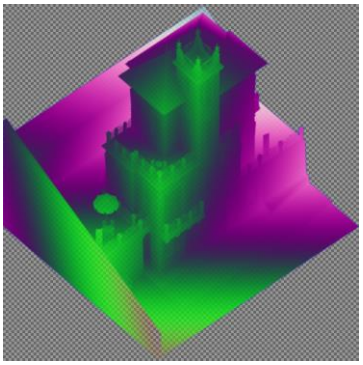
- Warp offset ξ_4 before storing it,
- Make the histogram more uniform,
- Solution independent of shadow map,
- Should map well to GPU instruction set,
- Experimentation leads to a scaled and shifted $\log \log \xi_4$.

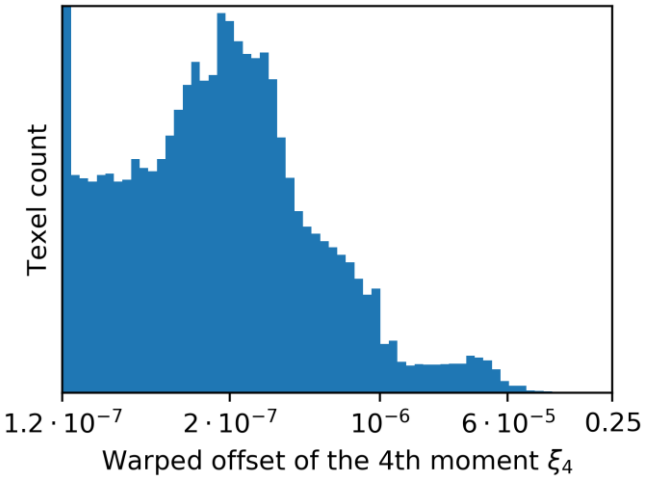
21

We should warp the offset before storing it to make this histogram more uniform. CLICK It is desirable to have a solution that works reasonably well independent of the present shadow map. It should also map well to the instruction set available in shaders. CLICK After some experimentation, we settled for the logarithm of the logarithm of the offset of the fourth moment. The growth properties of this transform provide a good idea of just how small the offset of the fourth moment tends to be.

Warped values are more uniform

Moment shadow map



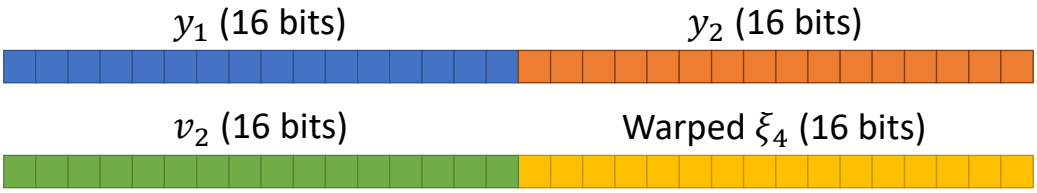


22

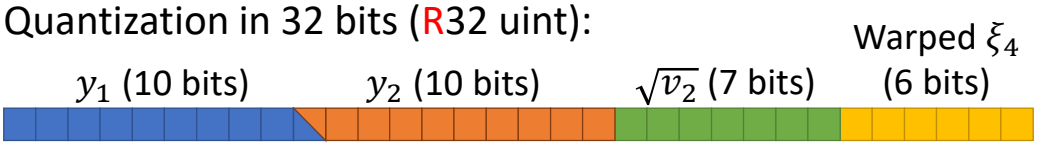
Indeed, this warp makes our histogram much more uniform. Now it can also be seen very clearly that nearly all texels have a value smaller than the rounding errors introduced by linear 64-bit quantization.

Quantization in 64 or 32 bits

Quantization in 64 bits (R16G16B16A16 normalized uint):



Quantization in 32 bits (R32 uint):



23

With this warp at hand, we are ready to define our quantized representations. We start with 64 bits where everything is quite straight-forward. We use a 16-bit normalized integer for each of the quantities. The only thing to remember is that the warp must be used for the offset of the fourth moment. CLICK At 32 bits, the scheme is a bit more intricate. We use 6 bits for the warped offset of the fourth moment. Seven bits are used for the weight but since this weight is often proportional to brightness, it is better to store the square root in the spirit of sRGB. Finally, we exploit that y_1 is always less than y_2 to gain one extra bit. Then each depth value can be stored with 10 bits of precision.

Optimized shading

- No need to convert back to 4 moments,
- Optimized algorithm computes shadow directly from $y_1, y_2, v_2, \xi_4,$
- See the paper.

24

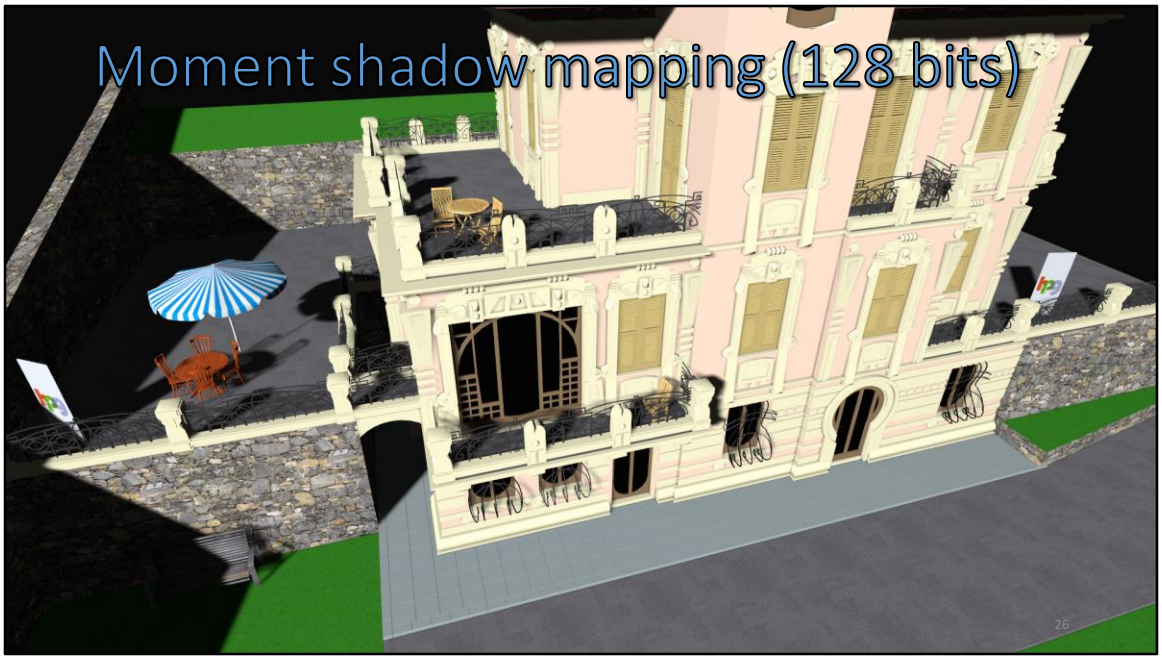
So now we know how to get a non-linearly quantized moment shadow map but what do we do with it? When we take a sample and want to shade a fragment, there is no need to convert the non-linear representation back to four moments. CLICK Instead, we can directly benefit from the non-linear representation to arrive at an optimized algorithm. CLICK For the rather mathematical details, you will have to read the paper.

Results

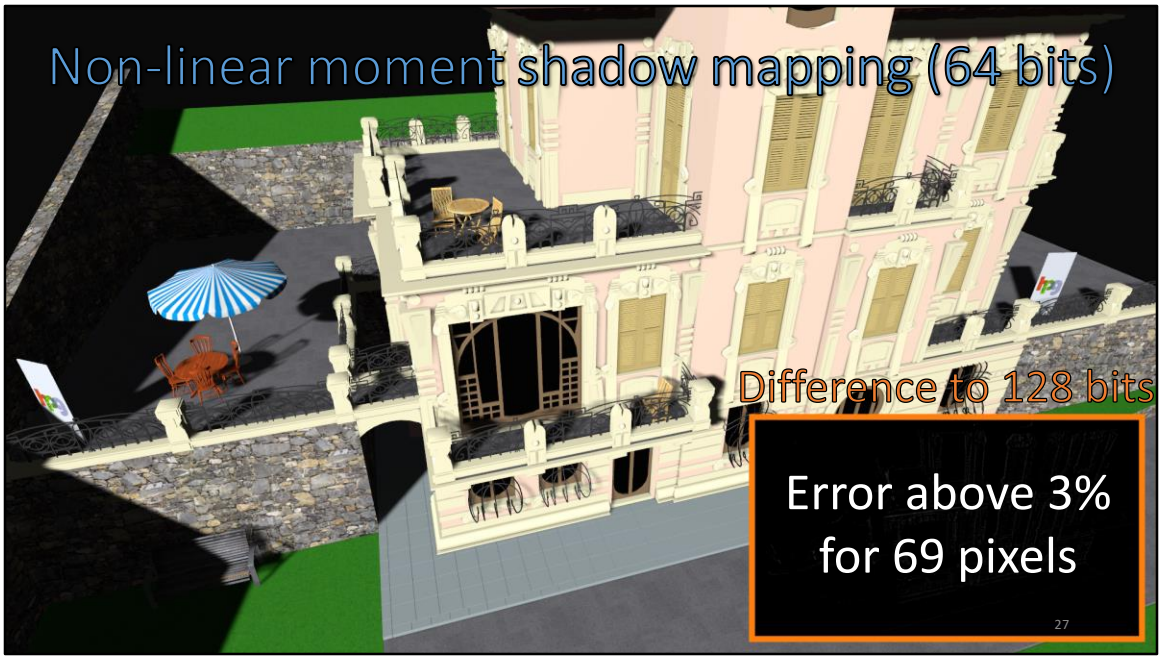
Of the non-linear quantization

25

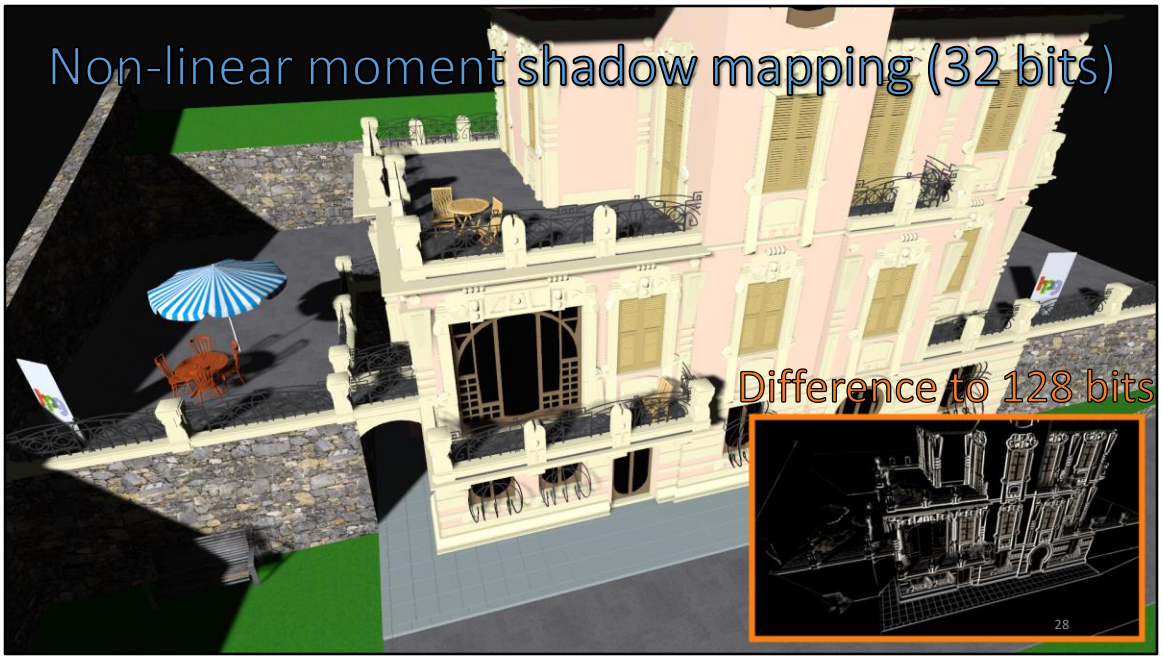
No lets take a look at the results of this quantization scheme.



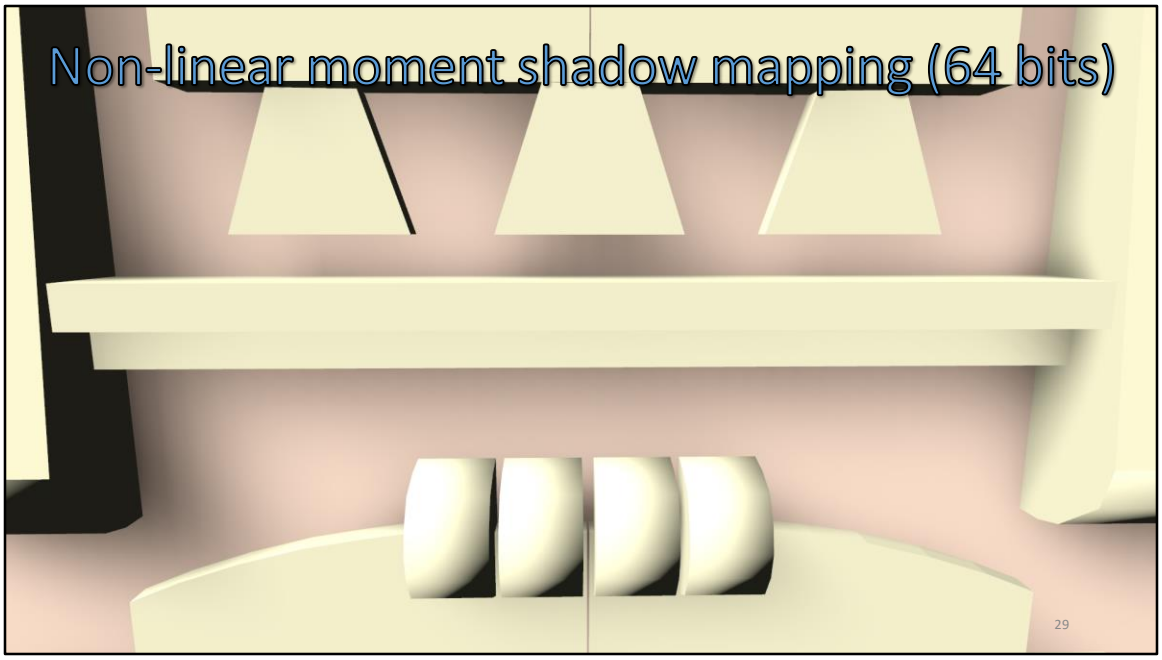
Here is a result of moment shadow mapping using one single-precision float per moment. As expected, this result looks good.



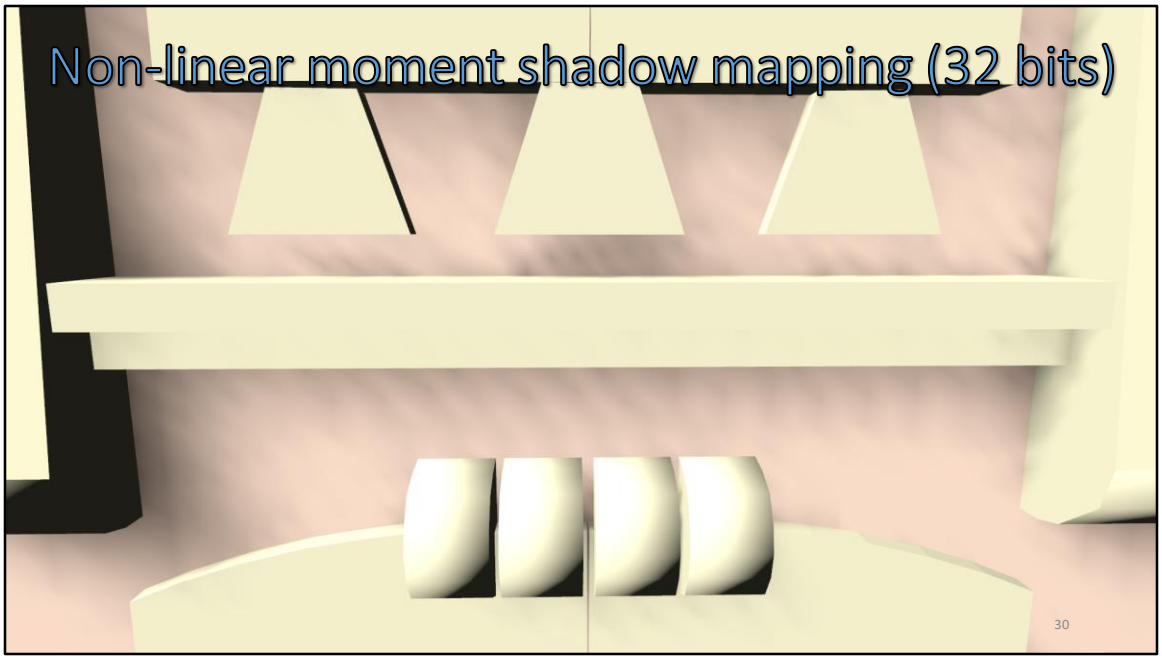
Now here is what we get with our non-linear quantization at 64 bits. The important thing to notice is that there is hardly anything to notice. CLICK Only 69 pixels have an absolute error in the shadow intensity above 3% and there is no case where this could be considered objectionable light leaking. For all practical purposes, linear quantization at 128 bits and non-linear quantization at 64 bits provide identical reconstruction quality.



Things are a little different if we look at our non-linear quantization at 32 bits. We get no significant light leaking but there are some artifacts along protrusions casting shadow over an extremely short range.



Lets take a closer look at that. Here is a close up showing non-linear moment shadow mapping at 64 bits per texel. The results are smooth and plausible.

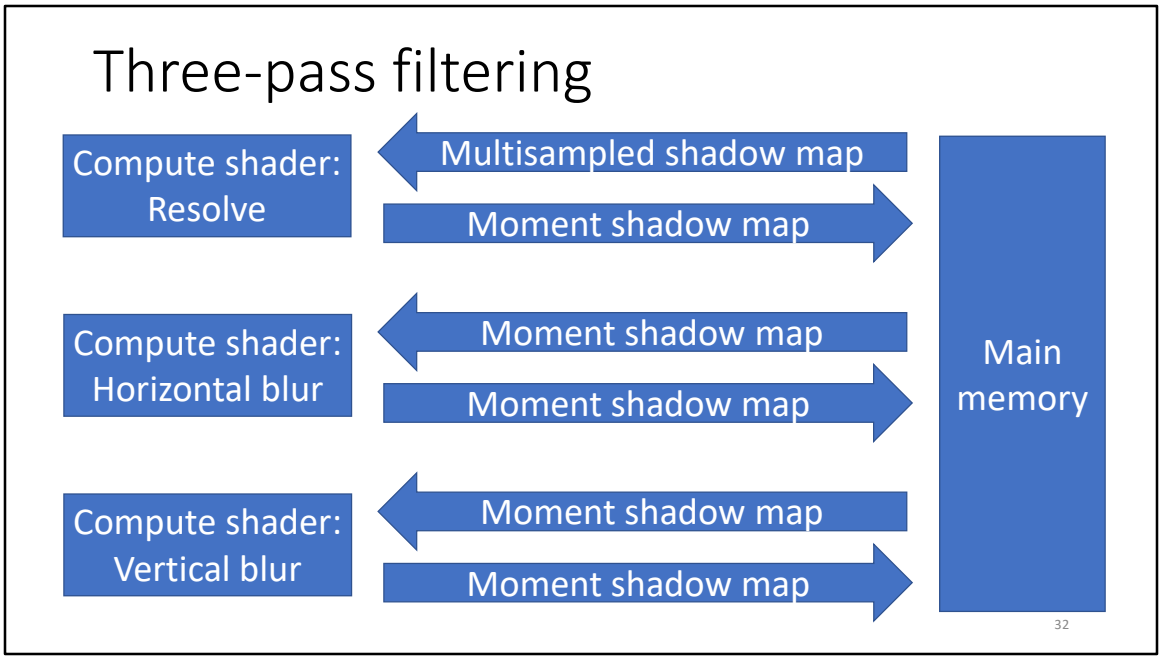


Here is what we get at 32 bits per texel. There are some rather obvious banding artifacts. These arise because 10 bits provide insufficient precision for the depth values when the shadows are cast over such short range. Dependent on the geometry of shadow casters and the depth range, this may or may not be a problem.

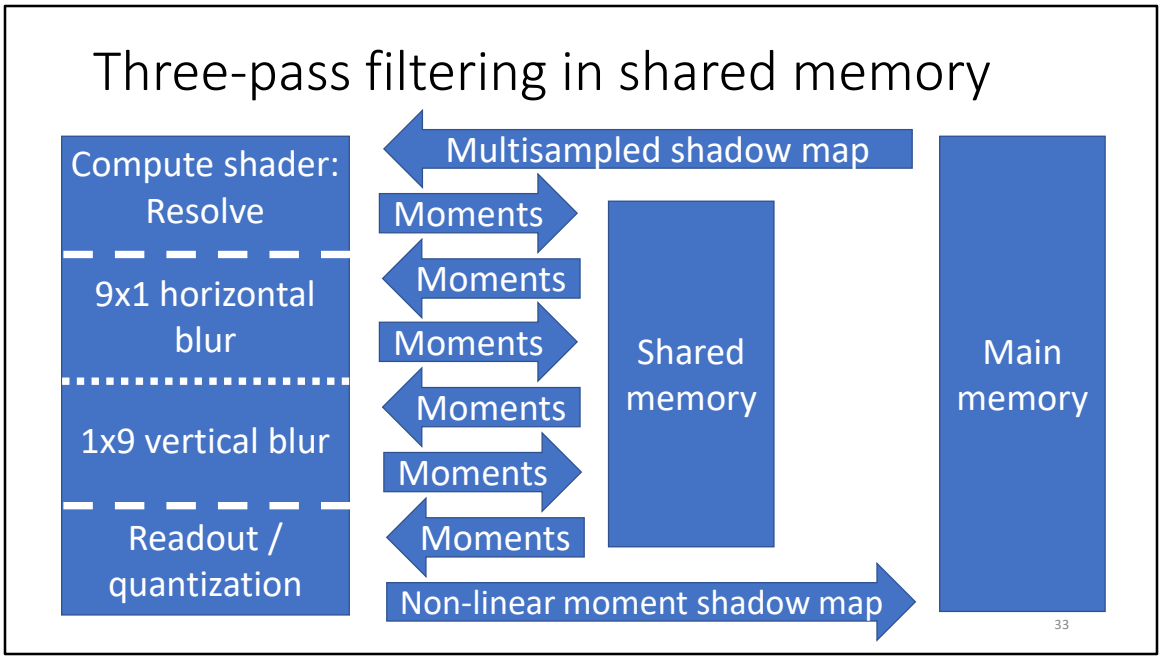
Filtering in shared memory

31

Next we will figure out how we can do the filtering for a non-linearly quantized moment shadow map efficiently.



Lets start by reviewing what was originally proposed for moment shadow maps. CLICK We start by reading samples from a multisampled shadow map in a compute shader. A vector of moments is generated for each sample and the results are combined in a resolve to obtain a moment shadow map which is then written back to main memory. CLICK Usually, we require additional filtering so we apply a separable blur. The horizontal pass reads the moment shadow map from main memory and writes it back. CLICK Finally, the same thing happens for the vertical pass. Presented like this, it is clear that there is a lot of redundant bandwidth usage. Since main memory bandwidth is the bottleneck, this is a major problem. Even at 64 bits per texel, the overhead is considerable but to avoid additional light leaking, we would need 128 bits per texel.



To really benefit from the non-linear quantization, we need something different. CLICK We start out doing the resolve in a compute shader, CLICK but this time we do not write the resulting moments back to main memory. Instead we store the results for a small block in shared memory. CLICK The horizontal blur reads the moments from there and writes them back there CLICK and so does the vertical blur. CLICK Finally, there is a readout step that grabs all the filtered moments and applies the non-linear quantization. CLICK Only then the results are written back to main memory. This way, we have reduced use of main memory to the bare minimum and since shared memory is much faster, it is no problem to use single-precision floats for the intermediate moments.

Accomplished design goals

- ✓ High occupancy,
- ✓ All 256 threads busy in all steps,
- ✓ Few redundant reads,
- ✓ No bank conflicts,
- ✓ Only two barriers.

34

Designing such a compute shader is tricky because a lot of constraints have to be met simultaneously. The solution that we provide with the paper accomplishes a high occupancy and all 256 threads of a thread group are busy in each step. At the same time, there are few redundant reads and no bank conflicts. Synchronization throughout the entire algorithm only requires two barriers.

Interpolation and dithering

35

Another problem that we encounter with the non-linear quantization is that hardware-accelerated bilinear interpolation becomes unavailable. We need to find an alternative.

Dithering replaces bilinear

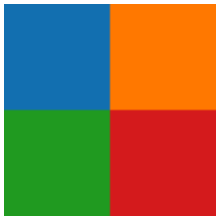
- Bilinear requires 4 samples → slow,
- Add random sub-texel offset to texture coordinate,
- Compute shadow for a single sample,
- Use precomputed blue noise.

36

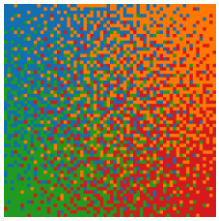
We could do the bilinear interpolation manually in the shader but processing four samples like this is quite expensive. CLICK Instead we choose to use dithering. We add a random offset to each texture coordinate before the lookup. Then we only compute the shadow using this one sample. CLICK To make the resulting noise as subtle as possible, we wish to have weak low frequencies. This is accomplished by using blue noise for the random offsets.

Blue noise dithering

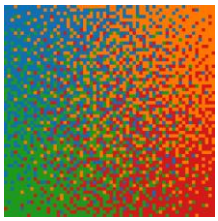
- 64 tileable textures at 64x64,
- Random texture and shift in each frame,



Nearest neighbor



Static dither



Animated dither

37

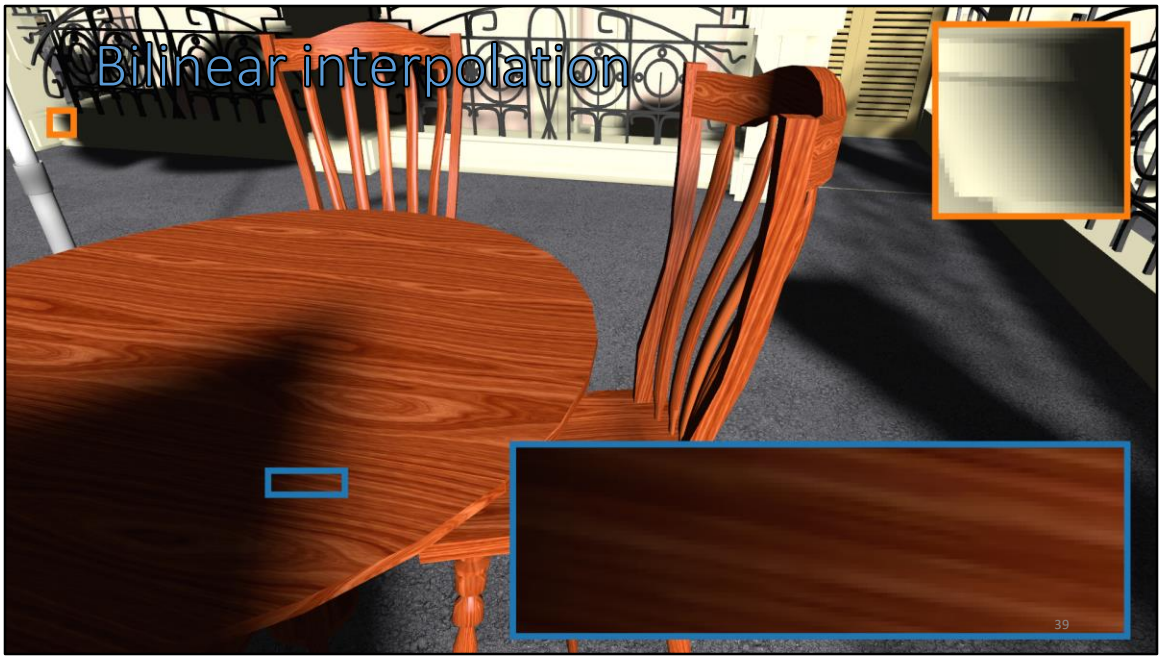
For this blue noise, we precompute 64 tileable textures at 64x64. In each frame, we tile the screen with a random texture using a random shift. One such texture is small enough to reside in L1 cache most of the time. This leads to an animated dither pattern such as the one shown here. It is far more pleasing than a static pattern or nearest neighbor interpolation.

Results

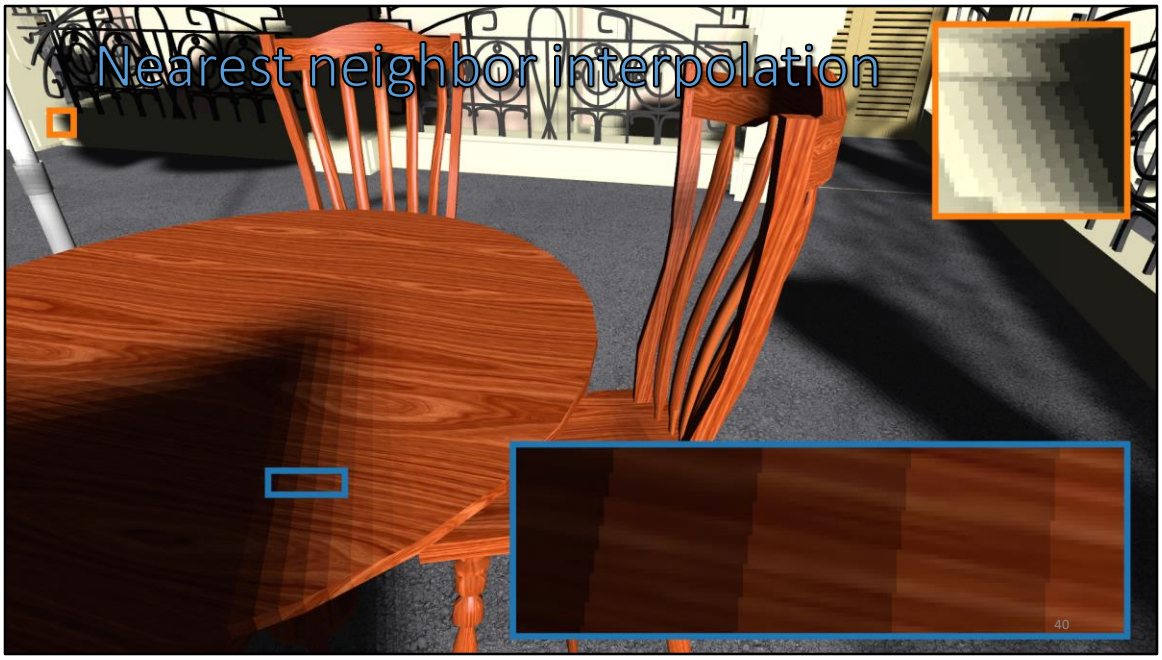
Of blue noise dithering

38

Again, we would like to take a look at the results.



Here is true bilinear interpolation. As expected, everything is looking good.



Nearest neighbor interpolation uses only a single sample so it is fast but the artifacts are too obvious to consider it as serious option.

Blue noise dithering

Play external video at 60 Hz.

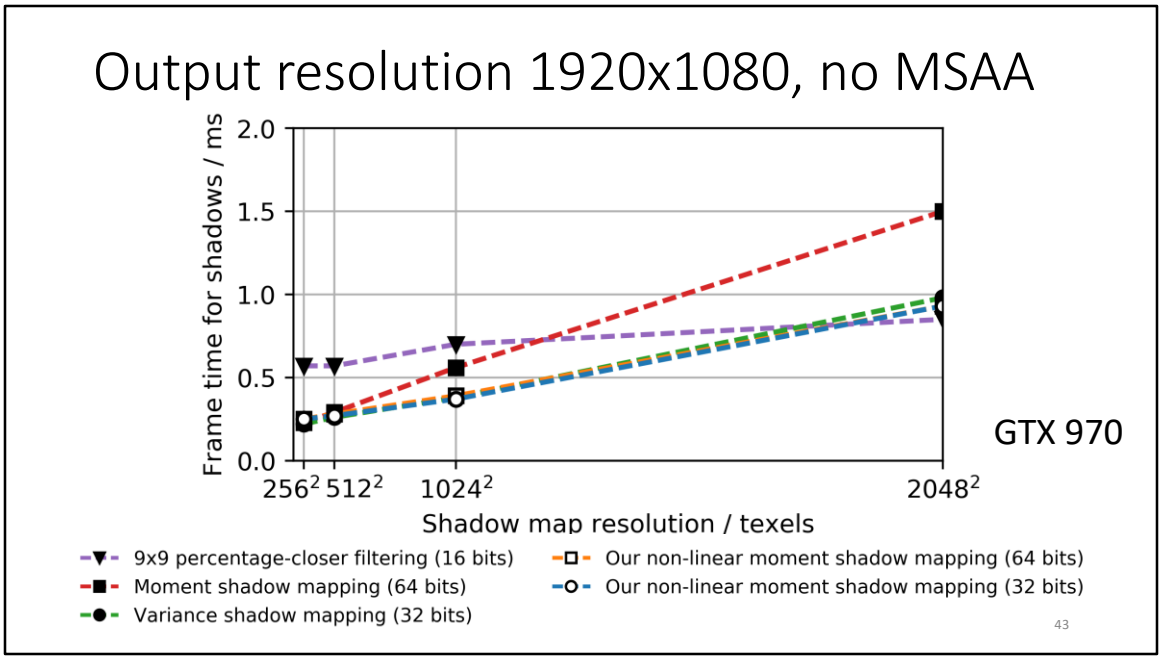
41

With blue noise dithering, we get a result that is perceptionally close to bilinear interpolation but using only a single sample. You can clearly see the dither patterns in the magnified insets but they are almost imperceptible without magnification. I do not claim that this solution will fit all cases but in a time where temporal antialiasing is widely used, it seems like a useful approach. Note that we do not use temporal antialiasing here.
Note: The supplementary video includes the video material used here.

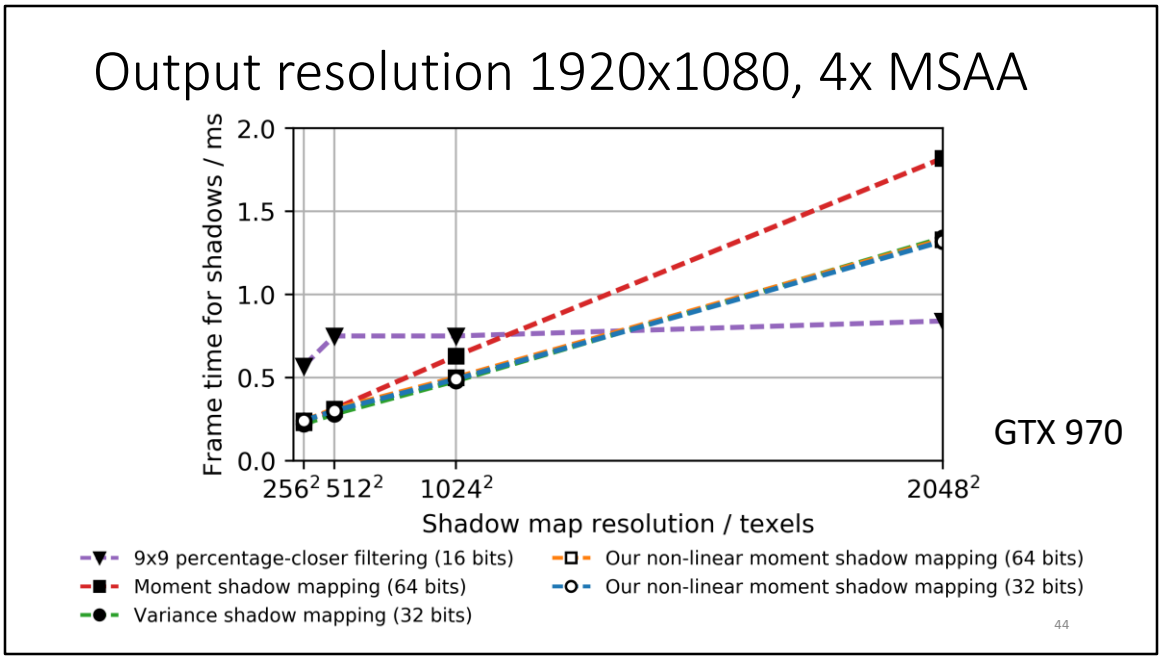
Run time

42

Finally, I would like to discuss the run time of our approach.

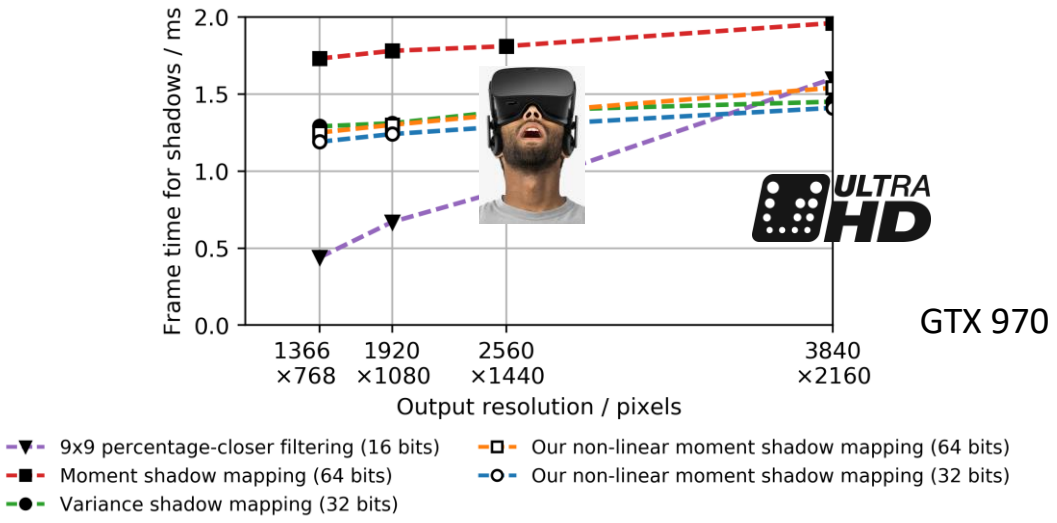


Here we fix the output resolution at full HD and do not use multisample antialiasing for the moment shadow maps. We consider the change of the frame time as the shadow map resolution increases. For percentage closer filtering, the cost grows slowly because a texel in the shadow map takes only 16 bits and does not require any post processing. For a variance shadow map the cost per texel is higher and for a moment shadow map at 64 bits it is still higher. CLICK As we add the graph for our non-linear quantization at 64 bits per texel, we note that the cost is almost identical to that of variance shadow mapping. CLICK However, going down to 32 bits does not yield a further improvement because we are now limited by shared memory bandwidth. Note that this speedup is due to the optimized filtering, not due to the quantization scheme. The quantization scheme serves to give us a quality improvement over linear quantization at 64 bits.



If we enable 4x multisample antialiasing for the moment shadow maps, the cost per texel increases and percentage-closer filtering without multisample antialiasing is faster for a 2048² shadow map. This extra cost is entirely due to the rendering of the shadow map itself, the cost for filtering does not grow with our optimized implementation. The reduction in shadow map aliasing is immense so the extra cost is well-justified.

Shadow map resolution 2048^2, 4x MSAA



Things get still more compelling if we keep the shadow map resolution fixed at 2048^2 with 4x MSAA and vary the output resolution. This time we see that the cost per pixel for 9x9 percentage-closer filtering is much higher than for moment shadow maps. Even at a 4k resolution, the overhead for shading is only 0.3 ms with all kinds of moment shadow maps. CLICK This is becoming more and more important as 4k displays are being pushed into the market. CLICK Also, head-mounted displays require high shading rates. CLICK As the hardware improves, these resolutions will only grow such that moment shadow mapping can lead to major savings especially with our novel non-linear quantization.

Conclusions

46

Lets wrap it up with some conclusions.

Conclusions

- Affordable antialiased shadows,
- Non-linear 64-bit quantization on par with 128 bits,
- 32-bit quantization only adds slight banding,
- Cost matches that of variance shadow mapping.

47

Non-linearly quantized moment shadow maps offer very affordable antialiased shadows. CLICK If we use 64 bits per texel, the quality is the same as with 128 bits. Thus, light leaking is minimal. CLICK At 32 bits per texel the only additional artifact is some banding in short-range shadows. Overall the quality of this approach is better than one might expect. CLICK In both cases, the run time cost matches that of variance shadow mapping.

Thanks!
Questions?

Contact me at Christoph@MomentsInGraphics.de.

48

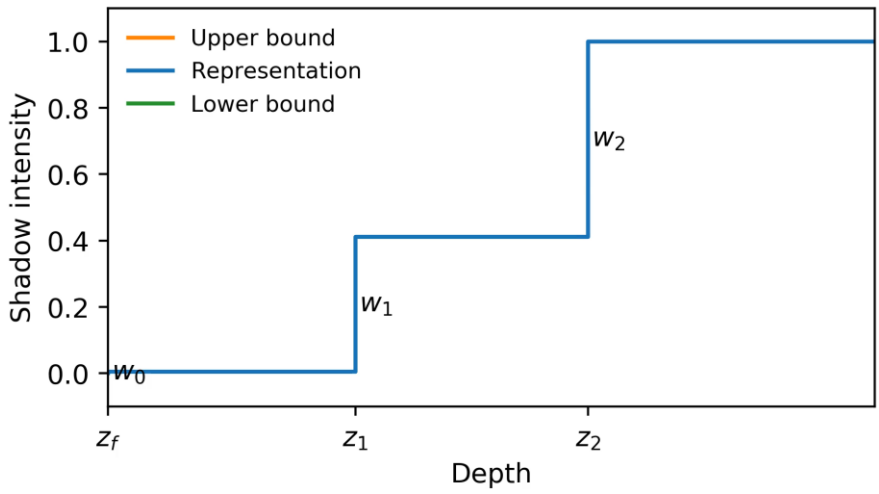
That’s all. If you have any questions about my work, please do not hesitate to contact me at Christoph@MomentsInGraphics.de. I am always curious to hear about uses of moment shadow mapping and if you run into any problems, there is a good chance that I will be able to help.

References

- [Peters15] Peters, C. and Klein, R. (2015). Moment shadow mapping. In Proceedings of the 19th ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games, i3D ’15, pages 7–14. ACM, doi: 10.1145/2699276.2699277.
- [Reeves87] Reeves, W. T., Salesin, D. H., and Cook, R. L. (1987). Rendering antialiased shadows with depth maps. In Proceedings of the 14th annual conference on Computer graphics and interactive techniques, SIGGRAPH ’87, pages 283–291. ACM, doi: 10.1145/37401.37435.
- [Williams78] Williams, L. (1978). Casting curved shadows on curved surfaces. In Proceedings of the 5th annual conference on Computer graphics and interactive techniques, SIGGRAPH ’78, pages 270–274. ACM, doi: 10.1145/800248.807402.

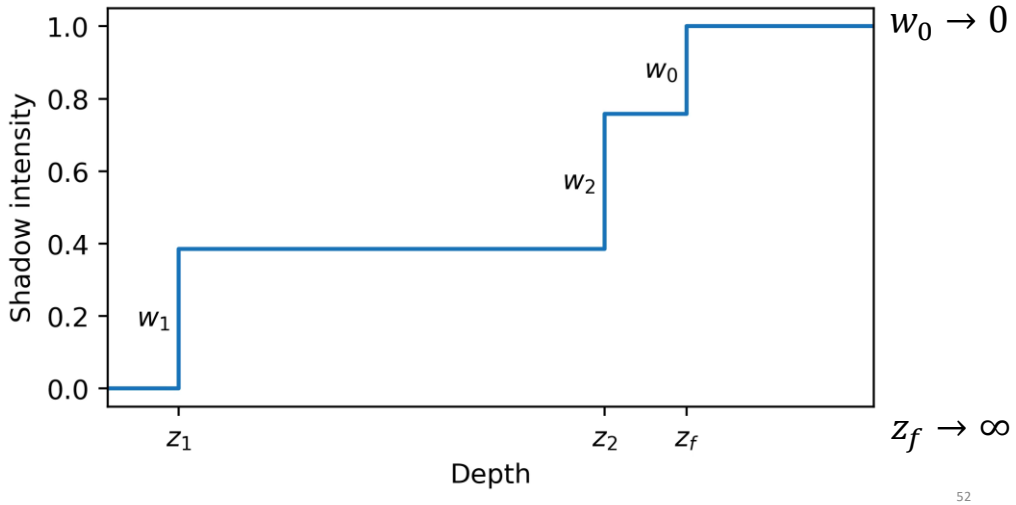
Appendix

Construction of the bounds



51

Taking z_f to the limit



52

Optimized shading

- Non-linear quantization gives us a matrix factorization ☺ ,

$$\begin{pmatrix} 1 & b_1 & b_2 \\ b_1 & b_2 & b_3 \\ b_2 & b_3 & b_4 \end{pmatrix} = v_1 \cdot \begin{pmatrix} 1 \\ y_1 \\ y_1^2 \end{pmatrix} \cdot \begin{pmatrix} 1 \\ y_1 \\ y_1^2 \end{pmatrix}^T + v_2 \cdot \begin{pmatrix} 1 \\ y_2 \\ y_2^2 \end{pmatrix} \cdot \begin{pmatrix} 1 \\ y_2 \\ y_2^2 \end{pmatrix}^T + \xi_4 \cdot \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \cdot \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}^T$$

$$= \begin{pmatrix} 1 & 1 & 0 \\ y_1 & y_2 & 0 \\ y_1^2 & y_2^2 & 1 \end{pmatrix} \cdot \begin{pmatrix} v_1 & 0 & 0 \\ 0 & v_2 & 0 \\ 0 & 0 & \xi_4 \end{pmatrix} \cdot \begin{pmatrix} 1 & 1 & 0 \\ y_1 & y_2 & 0 \\ y_1^2 & y_2^2 & 1 \end{pmatrix}^T$$

- We scale and shift to ensure $y_1 = 0$ and $y_2 = 1$,
- Solving a system of linear equations becomes effortless.

53