Democratic Tone Mapping Using Optimal K-means Clustering

Magnus Oskarsson(✉)

Centre for Mathematical Sciences, Lund University, Lund, Sweden
magnuso@maths.lth.se

Abstract. The field of high dynamic range imaging addresses the problem of capturing and displaying the large range of luminance levels found in the world, using devices with limited dynamic range. In this paper we present a novel tone mapping algorithm that is based on K-means clustering. Using dynamic programming we are able to, not only solve the clustering problem efficiently, but also find the global optimum. Our algorithm runs in $O(N^2K)$ for an image with N luminance levels and K output levels. We show that our algorithm gives comparable result to state-of-the-art tone mapping algorithms, but with the additional large benefit of a total lack of parameters. We test our algorithm on a number of standard high dynamic range images, and give qualitative comparisons to a number of state-of-the-art tone mapping algorithms.

Keywords: Image processing · Tone mapping · K-means clustering

1 Introduction

The human visual system can handle massively different levels in input brightness. This is necessary to cope with the large range of luminance levels that appear around us – for us to be able to navigate and operate in dim night light as well as in bright sun light. The field of high dynamic range (HDR) imaging tries to address the problem of capturing and displaying these large ranges using devices (cameras and displays) with limited dynamic range. During the last years the HDR field has grown and today many camera devices have built in functionality for acquiring HDR images. This can be done in hardware using sensors with pixels that can capture very large differences in dynamic range. It can also be done by taking several low dynamic range (LDR) images at different exposures and then combining them using software, [5,8,21]. One important part when working with HDR images is the ability to visualize them on LDR displays. The process of transferring an HDR image to an LDR image is known as tone mapping. Depending on the application the role of the tone mapping function can be different, but in most applications the ability to capture both detail in darker areas and very bright ones is important. Tone mapping can also be an important component in image enhancement for e.g. images taken under poor lighting [16,28]. We will in this paper present a new framework for doing

R.R. Paulsen and K.S. Pedersen (Eds.): SCIA 2015, LNCS 9127, pp. 354–365, 2015.
DOI: 10.1007/978-3-319-19665-7_29

automatic tone mapping without the need for manual parameter tuning. We will look at the tone mapping problem as a clustering problem. If we are given an input image with large dynamic range, i.e. with a large range of intensity values, we want to map these intensity values to a much smaller range. We can describe this problem as a clustering of the input intensity levels into a smaller set of output levels. In our setting we are looking at an HDR input with a very large input discretization and doing the clustering in three dimensions is not tractable. Iterative local algorithms will inevitably lead to local minima. Instead we work with only the luminance channel, and we show how we can find the global optimum using dynamic programming. This leads to a very efficient and stable tone mapping algorithm. We call our algorithm democratic tone mapping since all input pixels get to vote on which output levels we should use. An example of the output of our algorithm can be seen in Figure 1.

Fig. 1. The result of running our tone mapping algorithm on an HDR input image. We get a clear image, with details preserved and with lighting intact. There are no parameters that need to be set. HDR radiance map courtesy of P. Debevec [19].

1.1 Related Work

A large number of tone mapping algorithms have been proposed over the years. Some of the first work include [25,27]. One can divide the tone mapping algorithms into global algorithms, that apply a global transformation on the pixel intensities, and local algorithms, where the transformation also depends on the spatial structure in the image. For a discussion on the differences see [29]. The global algorithms include simply applying some fixed function such as a

logarithm or a power function. In [6] the authors present a method that adapts a logarithmic function to mimic the human visual system's response to HDR input. In [11] the image histogram is used and a variant of histogram equalization is applied but with additional properties based on ideas from human perception. Using histogram equalization will often lead to not efficiently using the colorspace, due to discretization effects.

The local algorithms usually apply some form of local filtering to be able to increase contrast locally. This often comes at the cost of higher computational complexity and can lead to strange artifacts. In [7] the authors use bilateral filtering to steer the local tone mapping. In [18] a perceptual model is used to steer the contrast mapping, which is performed in the gradient domain. In [17] the authors address the problem of designing display-dependent tone mappings. In [20] the authors propose an automatic version of the zone system developed by Ansel Adams for conventional photographic printing. The method also includes local filtering based on the photographic procedure of dodging and burning. The global part of their method is in spirit similar to our approach. In [12] they use K-means to cluster the image into regions and then apply individual gamma correction to each segment.

We address the problem of tone mapping as a clustering problem. This is not an entirely new idea in the realm of quantization of images. The idea of clustering color values was popular during the 1980's and 1990's, when the displays had very low dynamic range, and the object was to take an ordinary 24-bit color image and map it to a smaller palette of colors that could be displayed on the screen. In this case there have been a number of algorithms that use variants of K-means clustering, see [3,22,23]. Here the clustering was done on three dimensional input, i.e. color values. The algorithms used variants of the standard K-means [13] to avoid local minima. The number of input points was quite small and the number of output classes i.e. the palette, was relatively small so these methods worked well, but as shown in Section 5.1 they are prone to get stuck in local minima, when the size of the problem increases.

2 Problem Formulation

Let's consider the following problem. We are given an input gray value image, $I(x, y)$, with a large number, N, intensity levels. We would like to find an approximate image $\hat{I}(x, y)$ with a smaller amount, K, intensity levels, i.e. we would like to solve

$$\min_{c_1, c_2, \ldots, c_K} ||I - \hat{I}||_2, \tag{1}$$

where $I(x, y) \in \{u_1, u_2, \ldots, u_N\}$ and $\hat{I}(x, y) \in \{c_1, c_2, \ldots, c_K\}$. If we calculate the histogram corresponding to the input image's distribution we can reformulate the problem as:

Problem 1 (K-means clustering tone mapping). Given $K \in \mathbb{Z}^+$ and a number of gray values $u_i \in \mathbb{R}$ with a corresponding distribution histogram $h(i)$,

$i = 1, \ldots, N$, the K-means tone mapping problem is finding the K points $c_l \in \mathbb{R}$ that solve:

$$D(N, K) = \min_{c_1, c_2, \ldots, c_K} \sum_{i=0}^{N} h(i) d(u_i, c_1, \ldots, c_K)^2, \tag{2}$$

where

$$d(u_i, c_1, \ldots, c_K) = \min_l |u_i - c_l|. \tag{3}$$

This is a weighted K-means clustering problem. One usually solves it using some form of iterative scheme that converges to a local minimum. A classic way of solving it, is alternating between estimating the cluster centers c_l and the assignment of points u_i to clusters. If we have assigned n points u_i to a cluster l then the best estimate of c_l is the weighted mean

$$c_{\{1,\ldots,n\}} = \frac{\sum_{i=1}^{n} h(i) u_i}{\sum_{i=1}^{n} h(i)}. \tag{4}$$

For ease of notation we will henceforth use the notation c_l for cluster number l or $c_{\{1,\ldots,n\}}$ for the cluster corresponding to points $\{u_1, \ldots, u_n\}$. The contribution of this cluster to the error function (2) is then equal to:

$$f(u_1, \ldots, u_n) = \sum_{i=1}^{n} h(i)(u_i - c_{\{1,\ldots,n\}})^2. \tag{5}$$

The assignment that minimizes (2) given the cluster centers c_l is simply taking the nearest c_l for each point u_i. One can keep on iteratively alternating between assigning points to clusters and updating the cluster centers according to (4). It can easily be shown that this alternating scheme converges to a local minimum, but there are no guarantees that this is a global minimum. In fact for most problems it is highly dependent on the initialization. There are numerous ways of initializing. See [24] for an extensive review of K-means clustering methods.

The K-means clustering problem is in general NP-hard, for most dimensions, sizes of input and number of clusters, see [1,4,15,26] for details. However, since the points we are working with are one-dimensional i.e. $u_i \in \mathbb{R}$, we can actually find the global minimum of problem 1 using dynamic programming. This is what makes our method tractable. In the next section we will describe the details of our approach. We will in Section 4 describe how we use our solver to construct a tone mapping method for color images.

3 A Dynamic Programming Scheme

Problem 1 is a weighted K-means clustering problem, with data points in $\mathbb{R}$. We will now show how we can devise a dynamic programming scheme that accurately and fast gives the minimum solution to our problem. For details on dynamic programming see e.g. [10].

We use an approach similar to [2,26] and modify it to fit our weighted K-means problem. Since our data points are one-dimensional we can sort them in ascending order. Assume that we have obtained a solution $D(n,k)$ to (2) and let u_i be the smallest point that belongs to cluster k. Then it is clear that $D(i-1,k-1)$ is the optimal solution for the first $i-1$ points clustered into $k-1$ sets. This gives us the following recurrence relation for our problem:

$$D(n,k) = \min_{k \leq i \leq n} (D(i-1,k-1) + f(u_i,\dots,u_n)). \tag{6}$$

Equation (6) defines the Bellman equation for our dynamic programming scheme. and gives us our tools to solve problem 1. We iteratively solve $D(n,k)$ using (6) and store the results in an $N \times K$ matrix. The initial values for $n=1$ or $k=1$ are given by the trivial solutions. We can read out the optimal solution to our original problem at position (N,K) in the matrix. The clustering and the cluster centers of the optimal solution are then found by backtracking.

Algorithm 1. K-means clustering using dynamic programming

1: Given input points $\{u_1,\dots,u_N\}$, a distribution $h(i), i=1,\dots,N$ and K.
2: Iteratively solve $D(n,k)$ using (6) and (10) for $n=2,\dots,N$ and $k=2,\dots,K$.
3: Find the centers c_l, $l=1,\dots,K$ and the clustering by backtracking from the optimal solution $D(N,K)$.

In order to efficiently calculate $D(n,k)$ we need to be able to iteratively update the function f from (5). We do this in the following manner. We start by calculating the cumulative distribution $H(i)$ of $h(i)$, given by

$$H(i) = \sum_{j=1}^{i} h(j), \quad i = 1,\dots,N. \tag{7}$$

We can then update the weighted mean of a point set $\{u_1,\dots,u_n\}$ by:

$$c_{\{1,\dots,n\}} = \frac{h(n)u_n + H(n-1)\cdot c_{\{1,\dots,n-1\}}}{H(n)}. \tag{8}$$

Using this update we can also formulate an update to the error contribution of those points by:

$$f(u_1,\dots,u_n) = \sum_{i=1}^{n} h(i)(u_i - c_{\{1,\dots,n\}})^2 = \tag{9}$$

$$= \sum_{i=1}^{n-1} h(i)(u_i - c_{\{1,\dots,n-1\}})^2 + \frac{H(n-1)}{H(n)}(u_n - c_{\{1,\dots,n-1\}})^2 \cdot h(n). \tag{10}$$

To show this we can, without loss of generality, assume that we have transformed the coordinates so that $c_{\{1,\dots,n-1\}} = 0$ and hence $\sum_{i=1}^{n-1} h(i)u_i = 0$. This gives according to (8):

$$c_{\{1,\dots,n\}} = h(n)u_n/H(n). \tag{11}$$

Then

$$\sum_{i=1}^{n} h(i)(u_i - c_{\{1,\ldots,n\}})^2 = \sum_{i=1}^{n} h(i)(u_i - \frac{h(n)u_n}{H(n)})^2 = \tag{12}$$

$$= \sum_{i=1}^{n-1} h(i)(u_i - \frac{h(n)u_n}{H(n)})^2 + h(n)(u_n - \frac{h(n)u_n}{H(n)})^2 = \tag{13}$$

$$= \sum_{i=1}^{n-1} h(i)(u_i - \frac{h(n)u_n}{H(n)})^2 + h(n)u_n^2 \frac{H(n-1)^2}{H(n)^2} = \tag{14}$$

$$= \sum_{i=1}^{n-1} h(i)(u_i^2 - 2u_i \frac{h(n)u_n}{H(n)} + \frac{h(n)^2 u_n^2}{H(n)^2}) + h(n)u_n^2 \frac{H(n-1)^2}{H(n)^2} = \tag{15}$$

$$= \sum_{i=1}^{n-1} h(i)u_i^2 + \frac{H(n-1)h(n)^2 u_n^2}{H(n)^2} + h(n)u_n^2 \frac{H(n-1)^2}{H(n)^2} = \tag{16}$$

$$= \sum_{i=1}^{n-1} h(i)u_i^2 + \frac{H(n-1)h(n)u_n^2}{H(n)} \frac{(h(n) + H(n-1))}{H(n)} = \tag{17}$$

$$= \sum_{i=1}^{n-1} h(i)u_i^2 + \frac{H(n-1)h(n)u_n^2}{H(n)}. \tag{18}$$

Without using (10) each entry $D(n,k)$ would take n^2 iterations to calculate, and the total complexity would become $N \cdot K \cdot N^2 = N^3 K$. However using (10) we can compute $f(u_1, \ldots, u_n)$ in constant time, and this gives a total complexity of $N^2 K$.

4 Tone Mapping of Color Images

The discussion in the previous section was concerned with gray scale images. In this section we will describe the whole algorithm for an HDR color input image. We assume an RGB input image. Algorithm 1 is based on that the input points u_i are one-dimensional. There are a number of ways in which one could apply the clustering on a color image. One could run algorithm 1 on the R-,G- and B-channel independently or transfer the image into another color space such as HSV and do the processing on the V-channel. In order to have a fast, efficient and color-preserving method we have opted to run the algorithm on the luminance channel. We start by doing a preprocessing step by taking the logarithm of the RGB image, giving us I_{log}. We then estimate the luminance channel I_{gr} from I_{log}, using a standard weighted average. We can then run algorithm 1. When we have clustered the luminance channel into the desired K levels, we calculate our transfer function $F(s) : \{u_1, \ldots, u_N\} \rightarrow \{1, \ldots, K\}$ by finding the nearest neighbour of each input level s:

$$F(s) = \arg\min_l ||c_l - s||_2. \tag{19}$$

The output image is then constructed by applying the function F on the whole RGB-image pixel-wise. The different steps are sumarized in algorithm 2.

Algorithm 2. Democratic Tone Mapping (DTM)

1: Given a high bit color input image: I_{in}
2: Take the log to get $I_{log} = \log I_{in}$
3: Calculate the intensity channel I_{gr} of I_{log} .
4: Calculate the histogram $h(s)$ of I_{gr}.
5: Find the centers c_l using algorithm 1.
6: Estimate $F(s) : u_i \rightarrow c_l$ using nearest neighbours.
7: $I_{out} = F(I_{log})$.

5 Results

We have implemented our tone mapping algorithm and conducted a number of tests. In Section 5.1 we study the time complexity of our algorithm and then in Section 5.2 we show results on a set of standard HDR images and compare with a number of different tone mapping algorithms. We have consistently used $K = 256$ in our experiments, corresponding to 8-bit output, but this could be set to any output quantization you like.

5.1 Algorithm Complexity and Stability

A standard iterative K-means algorithm will converge to a local minimum. Algorithm 1 will converge to the global minimum, but one may ask how often the iterative scheme gets stuck in a local minimum, and how far this is from the global optimum. In order to investigate this we did a simple qualitative experiment where we ran our algorithm on an input image (with 2000 gray levels) and $K = 256$. We then ran the standard iterative K-means clustering and compared the resulting solution to the global optimum. We repeated this for a large number of runs with random initialization. In Figure 2a the results are shown. It shows a histogram over the L_2 differences between the local solution for the cluster centers and the true solution. The center points were of course sorted before the norm was taken. The figure clearly shows that in this case there was a large difference between the local solutions and the true solution, and in no case was the true optimum found using the local iterative method. Next we wanted to check if the total algorithm followed the expected complexity. In order to do this we ran algorithm 2 on an input image and varied the image size, the number of input gray levels N and the number of output gray levels K. Our implementation was done in Matlab, with the most time consuming step, i.e. the dynamic programming part, done using compiled mex-functions. The tests were conducted on a Mac mini with an 2.5 Ghz Intel core i5 processor. The results are shown

in Figure 2. In (b) we plot the total running time of algorithm 2 for different sizes of input images. One can see that the graph has a clear affine shape. This is in accordance with what would be expected. The dynamic programming part is independent of the image size, given that the number of input gray levels is fixed. The linear part comes from steps 2-4 and 7 of algorithm 2. In (c) and (d) the respective dependences on K and N are shown. The most time consuming part of the algorithm is the dynamic programming step, and this is linear in K and quadratic in N which is validated in the graph.

5.2 Results on HDR Images

We have tested our method on a number of HDR input images and compared with a number of standard tone mapping algorithms. The images were collected from [9,19]. We used the HDR image tool *Luminance HDR* [14] to do the processing. It contains implementations of a number of tone mapping algorithms.

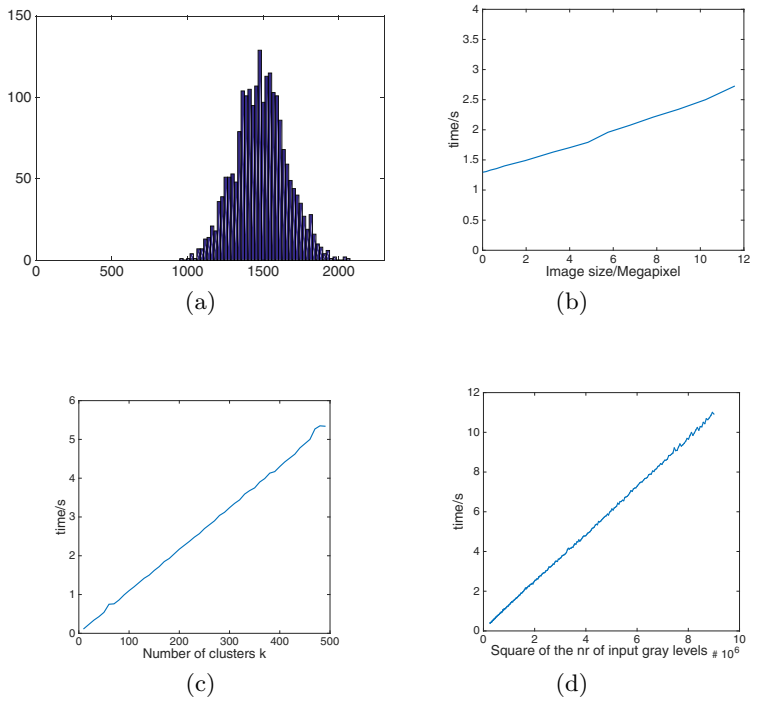

Fig. 2. (a) Histogram over the L_2-norms of the differences between the global optimum and local optimums after local optimization using random initialization. (b-d) Execution time for running the complete algorithm 2 as a function of (b) image size, (c) the output discretization K and (c) the square of the input discretization N. All plots follow the predicted behavior of the algorithm.

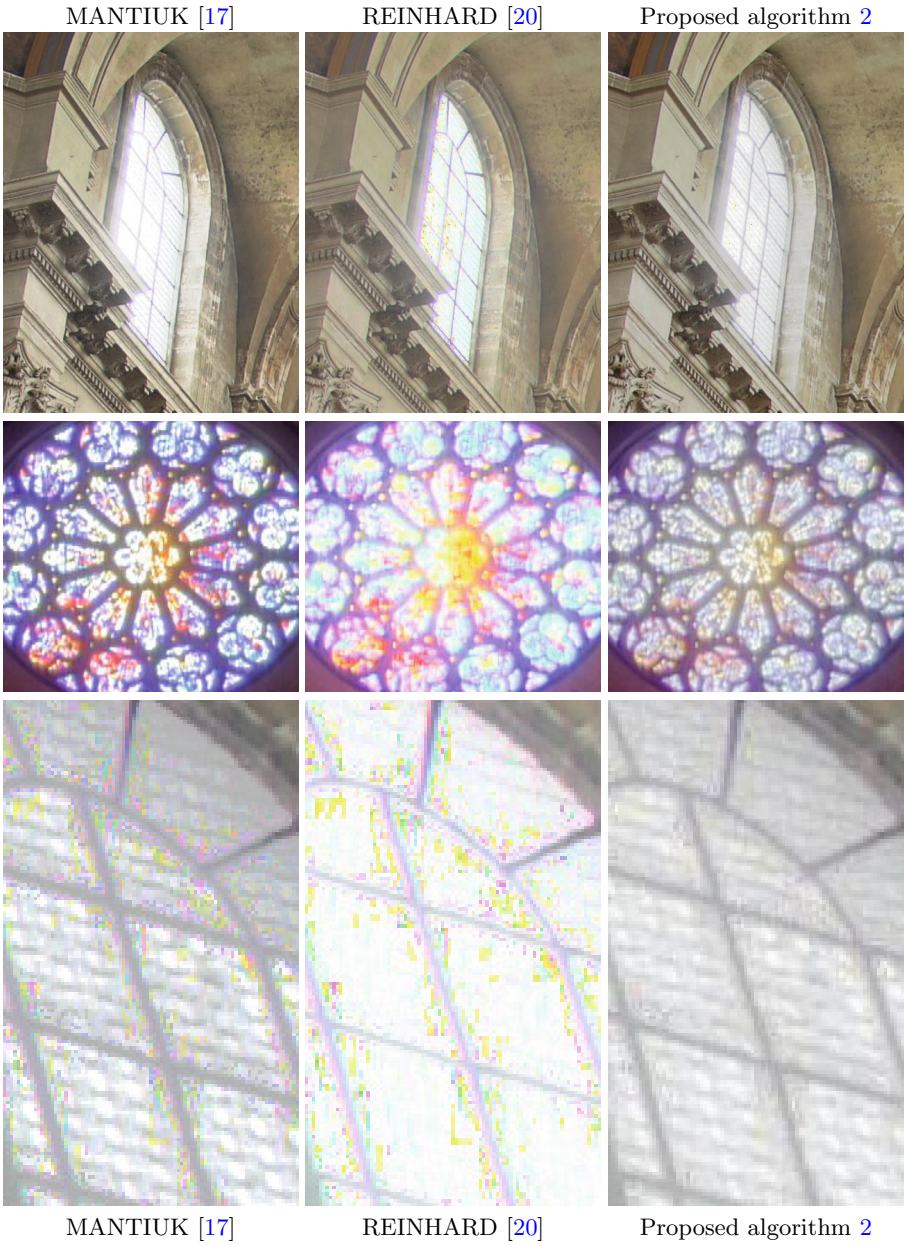

Fig. 3. The figure shows cutouts from the results on (top to bottom) NancyChurch3, Rosette and NancyChurch1. HDR radiance maps courtesy of R. Mantiuk [9] and P. Debevec [19]. The figure shows from left to right [17], [20] and our method. One can see that the compared methods suffer from over-saturation, color artifacts and loss of detail. The results are best viewed on screen.

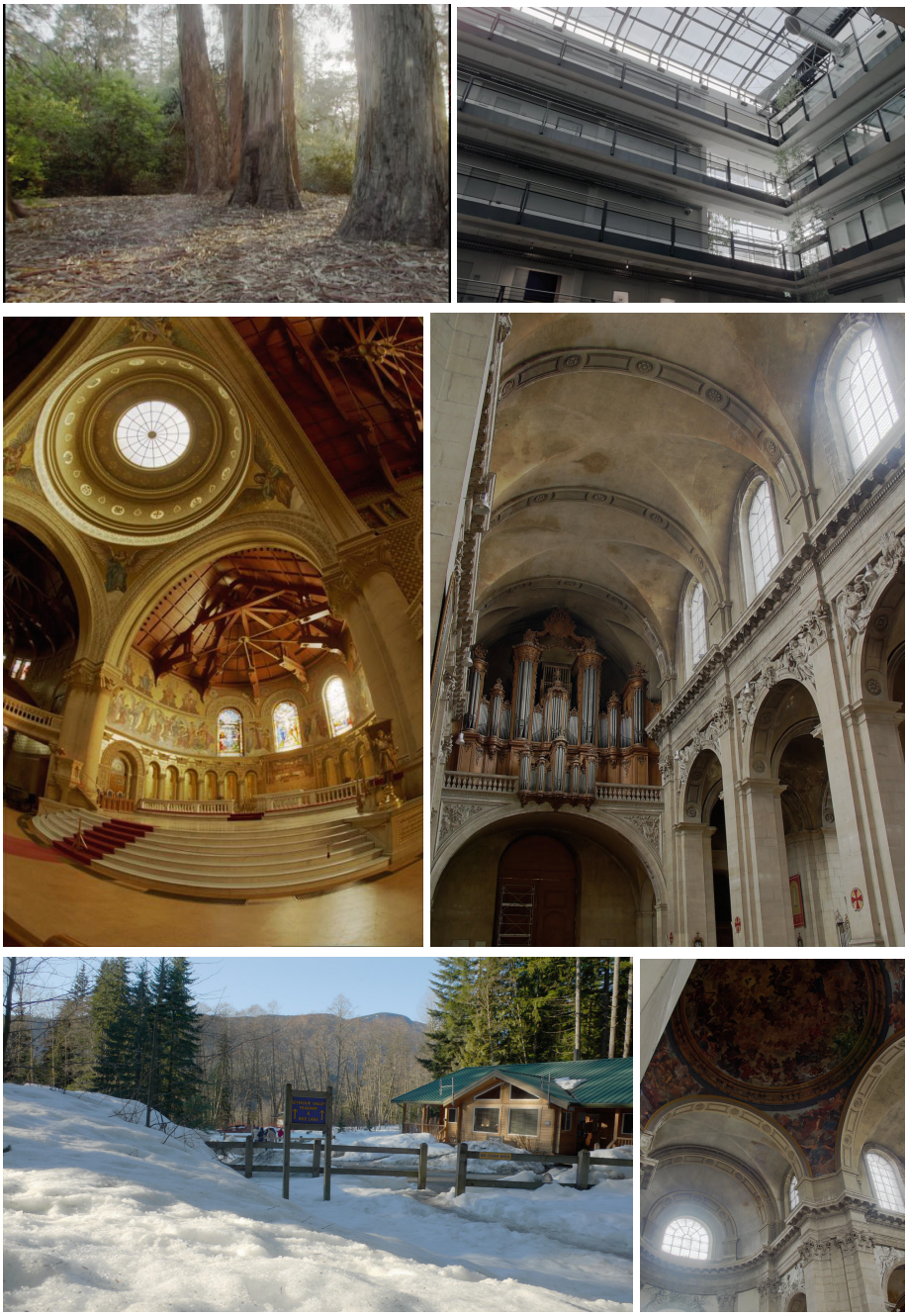

Fig. 4. The result of running our algorithm 2 on a number of HDR images No parameters need to be set to produce the output. The results are best viewed on screen. HDR radiance maps courtesy of P. Debevec [19] and R. Mantiuk [9].

We have throughout our tests only used the default parameter settings as supplied by *Luminance HDR*. It is probably so that in some cases better results can be found by tweaking the parameters manually, but since our method doesn't contain any parameters and the goal for us was to have an automatic system we opted for the default parameters. We have compared our method to the methods of Drago et al [6], Mantiuk et al [17,18], Reinhard et al [20] and Durand et al [7]. Of these we found that [17] and [20] gave significantly better results over the set of test images. Our method gave very similar results to these two methods.

In Figure 3 we show magnified cutouts from three example images. Here we can see that the two compared methods (on the left) exhibit problems with over-saturation, loss of detail resolution and color artifacts.

6 Conclusion

We have in this paper presented a novel tone mapping algorithm that is based on K-means clustering, using a dynamic programming approach. This enables us to not only solve the clustering problem efficiently but also find the global optimum. Our algorithm runs in $O(N^2K)$ for an image with N input luminance levels and K output levels, with comparable result to state-of-the-art tone mapping algorithms. One large benefit of our algorithm is its total lack of parameters. Some of the compared local methods can be tuned to get better effect in local color and contrast, but in many cases a totally automatic procedure is highly desirable. Our approach would also be quite straightforward to extend to HDR video, by working on a spatiotemporal block. Since the algorithm is linear in the number of pixels this would be very efficient.

Acknowledgments. This work was supported by ELLIIT and eSSENCE.

References

1. Aloise, D., Deshpande, A., Hansen, P., Popat, P.: Np-hardness of euclidean sum-of-squares clustering. Machine Learning **75**(2), 245–248 (2009)
2. Bellman, R.: A note on cluster analysis and dynamic programming. Mathematical Biosciences **18**(3), 311–312 (1973)
3. Celenk, M.: A color clustering technique for image segmentation. Computer Vision, Graphics, and Image Processing **52**(2), 145–170 (1990)
4. Dasgupta, S., Freund, Y.: Random projection trees for vector quantization. IEEE Transactions on Information Theory **55**(7), 3229–3242 (2009)
5. Debevec, P.E., Malik, J.: Recovering high dynamic range radiance maps from photographs. In: Proceedings of the 24th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH 1997, pp. 369–378. ACM (2007)
6. Drago, F., Myszkowski, K., Annen, T., Chiba, N.: Adaptive logarithmic mapping for displaying high contrast scenes. Computer Graphics Forum **22**(3), 419–426 (2003)
7. Durand, F., Dorsey, J.: Fast bilateral filtering for the display of high-dynamic-range images. ACM Transactions on Graphics (TOG) **21**(3), 257–266 (2002)

8. Grossberg, M.D., Nayar, S.K.: High dynamic range from multiple images: Which exposures to combine. In: Proc. ICCV Workshop on Color and Photometric Methods in Computer Vision (CPMCV), Nice, France (2003)
9. http://pfstools.sourceforge.net/hdr_gallery.html (Accessed 2014–09-01)
10. Kleinberg, J., Tardos, É.: Algorithm design. Addison-Wesley (2005)
11. Larson, G.W., Rushmeier, H., Piatko, C.: A visibility matching tone reproduction operator for high dynamic range scenes. IEEE Transactions on Visualization and Computer Graphics **3**(4), 291–306 (1997)
12. Lee, J.W., Park, R.H., Chang, S.: Local tone mapping using the k-means algorithm and automatic gamma setting. IEEE Transactions on Consumer Electronics **57**(1), 209–217 (2011)
13. Lloyd, S.: Least squares quantization in pcm. IEEE Transactions on Information Theory **28**(2), 129–137 (1982)
14. http://qtpfsgui.sourceforge.net (Accessed: 2014–09-01)
15. Mahajan, M., Nimbhorkar, P., Varadarajan, K.: The planar k-means problem is NP-hard. In: Das, S., Uehara, R. (eds.) WALCOM 2009. LNCS, vol. 5431, pp. 274–285. Springer, Heidelberg (2009)
16. Malm, H., Oskarsson, M., Warrant, E., Clarberg, P., Hasselgren, J., Lejdfors, C.: Adaptive enhancement and noise reduction in very low light-level video. In: IEEE 11th International Conference on Computer Vision, 2007. ICCV 2007, pp. 1–8. IEEE (2007)
17. Mantiuk, R., Daly, S., Kerofsky, L.: Display adaptive tone mapping. ACM Transactions on Graphics (TOG) **27**(3), 68:1–68:10 (2008)
18. Mantiuk, R., Myszkowski, K., Seidel, H.P.: A perceptual framework for contrast processing of high dynamic range images. ACM Transactions on Applied Perception (TAP) **3**(3), 286–308 (2006)
19. http://www.pauldebevec.com/Research/HDR (Accessed: 2014–10-01)
20. Reinhard, E., Stark, M., Shirley, P., Ferwerda, J.: Photographic tone reproduction for digital images. ACM Transactions on Graphics (TOG) **21**(3), 267–276 (2002)
21. Robertson, M.A., Borman, S., Stevenson, R.L.: Dynamic range improvement through multiple exposures. In: Proceedings International Conference on Image Processing, ICIP 99, Kobe, Japan. vol. 3, pp. 159–163. IEEE (1999)
22. Scheunders, P.: A comparison of clustering algorithms applied to color image quantization. Pattern Recognition Letters **18**(11), 1379–1384 (1997)
23. Scheunders, P.: A genetic c-means clustering algorithm applied to color image quantization. Pattern Recognition **30**(6), 859–866 (1997)
24. Steinley, D.: K-means clustering: a half-century synthesis. British Journal of Mathematical and Statistical Psychology **59**(1), 1–34 (2006)
25. Tumblin, J., Rushmeier, H.: Tone reproduction for realistic images. Computer Graphics and Applications, IEEE **13**(6), 42–48 (1993)
26. Wang, H., Song, M.: Ckmeans. 1d. dp: optimal k-means clustering in one dimension by dynamic programming. The R Journal **3**(2), 29–33 (2011)
27. Ward, G.: A contrast-based scalefactor for luminance display. Graphics gems IV, pp. 415–421 (1994)
28. Warrant, E., Oskarsson, M., Malm, H.: The remarkable visual abilities of nocturnal insects: Neural principles and bioinspired night-vision algorithms. Proceedings of the IEEE **102**(10), 1411–1426 (2014)
29. Wilkie, K., Devlin, A., Chalmers, A., Purgathofer, W.: Tone reproduction and physically based spectral rendering. Eurographics 2002: State of the Art Reports, pp. 101–123 (2002)