Two-scale tone management for photographic look

Daniel Yim

Soonmin Bae, Sylvain Paris, and Frédo Durand. Two-scale tone management for photographic look. In *SIGGRAPH '06: ACM SIGGRAPH 2006 Papers*, pages 637–645, New York, NY, USA, 2006. ACM

Bae et al.'s Two-scale Tone Management for Photographic Look presents a method for manipulating tone in images. Previous works in tone management focused on techniques to directly manipulate tone; this paper focuses on how tone affects the "look" of an image and on finding a way to transfer the "look" from one image to another. The authors make the observation that the primary factors that influence the "look" of an image are the spatial distribution of tones and the amount of texture. This observation inspired the approach of splitting an image into two parts: the base layer, which handles transferring large-scale effects, and the detail layer, which handles the texture distribution.

To split the image into the two layers, a bilateral filter is used. Given the input image I, we set $B = bf(I)$ and $D = I - B$ where B and D is the base layer and the detail layer respectively. $bf(I)$ is the bilateral filter. It is defined as: $bf(I)_p = \frac{1}{k}\sum_{q\in I} g_{\sigma_s}(||p-q||)g_{\sigma_r}(|I_p - I_q|)I_q$ where $k = \sum_{q\in I} g_{\sigma_s}(||p-q||)g_{\sigma_r}(|I_p - I_q|)$. Good values for the parameters σ_s and σ_r for the Gaussian functions in the bilateral filter are $min(width, height)/16$ and the 90th percentile of $||\nabla I||$ respectively. These values seemed to have been found by using trial and error. This process could potentially reverse the gradient. To correct this, first a gradient field is created with $v = (x_v, y_v)$ where $x_v = \delta D/\delta x$but clipped to the range $[0, |\delta I/\delta x|]$ in the direction of $\delta I/\delta x$ and with y_v defined in a similar manner. This gradient field is then used to solve the corresponding Poisson equation to get the corrected D. B is updated so that $B = I - D$.

Histogram matching is used to transfer the histogram of the tone from the model image, M, onto B. Because B contains the large-scale spatial distribution of tones, this effectively transfers the large-scale global contrasts. Handling local contrast/texture is not as easy. The problem is that the amount of texture is often not uniform and can vary across a photo (e.g. smooth skies vs. textured forest). The authors define $T(I)$ to be the "textureness" of an image. Calculating textureness requires doing a high-pass filter over the image, then using a cross-bilateral filter on the high-pass image which uses the intensity image for the edge-preserving term in the bilateral filter. To transfer textureness from the model, textureness maps, $T(I)$ and $T(M)$ is calculated. The histogram of $T(M)$ is transferred to $T(I)$ to make T'. The texture scale factor, ρ, is defined per pixel p as $\rho_p = max(0, \frac{T'_p - T(B)_p}{T(D)_p})$.

The detail and base layers are combined by the equation: $O = B + \rho D$. Because ρ is defined per pixel, each pixel can get different multiplicative factors that vary the amount of detail spatially. Further post-processing is required on the output, O, to bring the image intensities within displayable range. The authors also solve another Poisson equation that prevents the gradient in the final image from varying too much from the original image.

While many of the tools used in this paper are not new (e.g. Poisson image editing, bilateral filtering to preserve edges, histogram matching), the authors did make small improvements to existing techniques such as preventing gradient reversal and using cross-bilateral filtering on the high-pass image to preserve edges but these seem like relatively minor improvements. The interesting contribution of this paper comes from its focus on creating a stylistic "look" and designing the whole pipeline process to accomplish this. It was also impressive that all of this can be automated, but still allow user interaction afterwards to refine the results. Another good aspect of this technique is that it is easy to use; the only required inputs are the source image and the model image.

One disadvantage mentioned by the authors is that this technique does not lend itself well to JPEG files because of noise and JPEG artifacts in the input. Transferring the texture in this case can amplify the defects. Although RAW formats might be popular in DSLR cameras, JPEG is still a very popular format in point-and-shoot cameras. Similarly, this technique can have troubles with videos because individual frames can contain large amounts of fluctuation due to auto-exposure, autofocus, and motion blurs.

The authors mention another disadvantage to their technique: it does not work well with portraits, especially when the model image is something completely unrelated to portraits. The main reason is that undesired skin defects are amplified. There can be scenarios though, where the user might want to transfer the stylistic changes to the background, even in a portrait. Perhaps it could be possible to extend the algorithm to specify regions where the texture detail is not applied (e.g. the face) to avoid the unwanted amplification of skin texture.