Adaptive user interfaces for relating high-level concepts to low-level photographic parameters

Edward Scott, Pubudu Madhawa Silva, Bryan Pardo, and Thrasyvoulos N. Pappas

EECS Department, Northwestern University, 2145 Sheridan Rd., Evanston, IL 60201, USA

ABSTRACT

Common controls for photographic editing can be difficult to use and have a significant learning curve. Often, a user does not know a direct mapping from a high-level concept (such as "soft") to the available parameters or controls. In addition, many concepts are subjective in nature, and the appropriate mapping may vary from user to user. To overcome these problems, we propose a system that can quickly learn a mapping from a high-level subjective concept onto low- level image controls using machine learning techniques. To learn such a concept, the system shows the user a series of training images that are generated by modifying a seed image along different dimensions (e.g., color, sharpness), and collects the user ratings of how well each training image matches the concept. Since it is known precisely how each modified example is different from the original, the system can determine the correlation between the user ratings and the image parameters to generate a controller tailored to the concept for the given user. The end result – a personalized image controller – is applicable to a variety of concepts. We have demonstrated the utility of this approach to relate low-level parameters, such as color balance and sharpness, to simple concepts, such as "lightness" and "crispness," as well as more complex and subjective concepts, such as "pleasantness." We have also applied the proposed approach to relate subband statistics (variance) to perceived roughness of visual textures (from the CUReT database).

Keywords: Machine learning, image concept, perceived roughness, support vector regression.

1. INTRODUCTION

Common tools for photographic editing can be difficult to use and have a significant learning curve. Typically, there is no direct mapping from a high-level concept that the user has in mind, such as "soft" or "pleasant," to the available low-level parameters or controls. In addition, many concepts are subjective in nature, and the appropriate mapping may vary from user to user. To overcome these problems, we propose a system that can quickly learn a mapping from a high-level subjective concept onto low-level parameters using machine learning techniques. To learn such concepts, the system shows the user a series of training images that are generated by modifying a seed image along different dimensions (e.g., color, sharpness), and collects the user ratings of how well each training image matches the concept. Since the low-level transformations used to generate the training images are known, the system can determine the mapping between the user ratings and the low-level parameters to generate a controller for the given concept and user. The end result – a personalized image controller – is applicable to a variety of concepts.

We propose an adaptive user interface that enables the user to quickly define a controller for a high level concept based on the available low-level controls. Such interfaces have been successfully applied in the audio domain to the problems of equalization by Sabin and Pardo[1] and reverberation by Rafii and Pardo.[2] Yang and Peng[3] proposed a system that can transfer the "mood" of one image to another by matching the color palettes of the two images. Their work is similar to ours in the sense that it assumes a mapping of high level image concepts ("moods") to low level parameters, but it is restricted to a set of 24 moods, and there is no ability or attempt to customize the mappings to specific users. In contrast, the adaptive user interface we have developed can learn a mapping for nearly any concept a user wishes to teach it, provided that the concept may be represented by the systems parameters. To our knowledge, this is the first application of adaptive user interfaces to image editing.

In Refs. 1 and 2, the user rates a number of audio examples based on how well each example corresponds to some audio concept or descriptor, where each example is a version of one original sample modified in the relevant

Further author information: Send correspondence to T.N.P.: E-mail: pappas@eecs.northwestern.edu

Human Vision and Electronic Imaging XVI, edited by Bernice E. Rogowitz, Thrasyvoulos N. Pappas,
Proc. of SPIE-IS&T Electronic Imaging, SPIE Vol. 7865, 78650A · © 2011 SPIE-IS&T
CCC code: 0277-786X/11/$18 · doi: 10.1117/12.879072

dimensions (in this case either frequency content or reverberation). We adopt the same approach in the studies we present in this paper. It is important to point out that the only requirement for the success of the proposed approach is that there is some type of correlation between the low-level controls and the high-level concept.

The most significant and commonly used parameters in image editing interfaces relate to color balance. Another dimension that is relatively straightforward to manipulate is image sharpness. Other effects include quantization, noise reduction (or addition), posterization, and cropping. To demonstrate the utility of the proposed approach, we have thus selected color balance and sharpness as the low-level image parameters. We have also selected high-level concepts that are affected by these parameters. We have selected both simple concepts, such as "lightness" and "crispness," as well as more complex and subjective concepts, such as "pleasantness." The simple concepts are more or less directly related to the selected low-level parameters, lightness to color luminance and crispness to image sharpness and color contrast, which in turn depends on color balance. The mapping from low-level parameters to pleasantness is not as obvious. However, even for the simple concepts, the mapping is not trivial, and it is interesting to explore.

One of the most important aspects of the proposed approach for developing an adaptive user interface is how the training data – in this case the user ratings – are collected. Every example presented to the user for rating is a modified version of an original image along the dimensions of interest, e.g., color balance and sharpness. All other variables beyond the control of the system are eliminated, thus enabling the system to learn a mapping for the concept based only on the relevant parameters.

Another use of the proposed approach is for the study of high-level concepts that may not have an immediate/obvious application in image editing interfaces, but are important for image analysis and understanding, and content-based retrieval. One such example is the study of human perception of roughness of visual textures. Van Egmond, *et al.*[4] studied the subjective roughness of images from the "CUReT" database[5] and found that roughness judgments were systematic and subjects could differentiate among images. However, little is known about the image parameters that affect roughness. Van Egmond, *et al.*[6] proposed a simple objective predictor of subjective visual roughness based on the variance of the coefficients of a subband decomposition. Given the fact that the variances of the subband coefficients affect the perception of image roughness, an interesting application of the user rating collection method described above is to figure out the mapping between the variance of the subband coefficients and image roughness. Although the end goal is not a controller for visual roughness, the concept of an adaptive user interface still applies in the sense that the users are teaching the system how to measure perceived roughness of visual textures.

The paper is organized as follows. In Section 2 we present the proposed adaptive user interface and apply it to simple and complex concepts. In Section 3 we apply the proposed techniques to the perception of visual roughness. The concluding remarks are presented in Section 4.

2. ADAPTIVE USER INTERFACE FOR IMAGE EDITING

In this section, we describe the system for learning a mapping from a high-level subjective concept onto a set of low-level parameters using machine learning techniques. First, we have to select the high-level concept, the set of parameters, and the range over which each parameter will be varied. The quantization of the parameter space must be carefully selected to result in perceptually significant changes in the image.

The proposed system consists of four phases, as shown in Fig. 1. For illustration purposes, in this figure, the high-level concept is softness and the low-level parameters are the three coordinates of CIELAB color space (L^*, a^*, and b^*) and a sharpness parameter s. In the *first* phase, the system generates 60 examples of an image, each modified by a different degree in each of the four dimensions. In the *second* phase, the system collects training data for the 60 examples from the user. The user labels each example with a rating in the $[-1, 1]$ range (using a slider) that represents how well the example corresponds to the concept that the system is trying to learn. In the *third* phase, the system learns the mapping between the user ratings and each of the parameters. In the *fourth* phase, given a user rating (using the same slider and range of values $[-1, 1]$) and the learned model of the concept, the system calculates the parameter values that will generate an image with the given rating. As we discussed, the system is only expected to work for concepts which may somehow be expressed by the low-level parameters. We now take a more careful look at the parameter space.

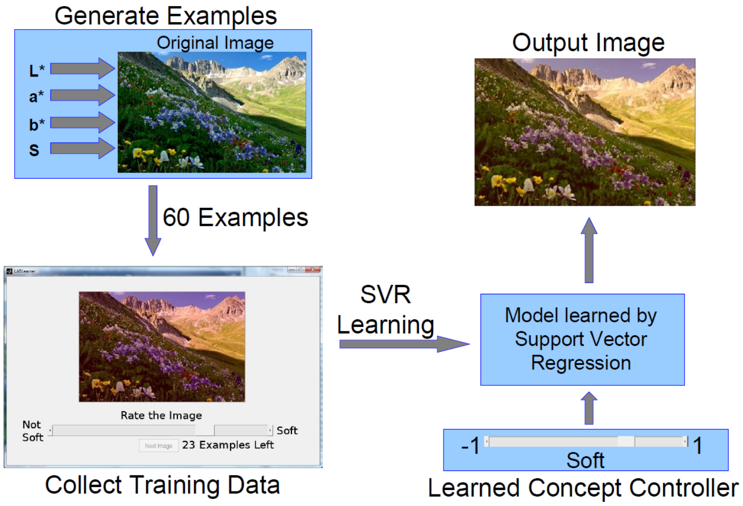


Figure 1. The system work flow

2.1 Defining the Parameter Space

We selected four parameters targeted at obtaining the maximum impact on the appearance of a photograph. The first three parameters, d_L, d_a, and d_b, represent adjustments of the L^*, a^*, and b^* values of the image pixels in CIELAB space. The fourth parameter, d_s, represents image sharpening if its value is positive and image blurring if its value is negative. We chose the CIELAB space (as opposed to RGB or HSV space) because it separates luminance (L^*) and chrominance (a^* and b^*) values and because it is intended to be perceptually uniform, in that equal steps in parameter space represent equal steps in perceived color differences. Although CIELAB is in fact only perceptually uniform over small distances, it is still better in this respect than RGB or HSV space.

One issue that became apparent in deciding how to modify the L^*, a^*, and b^* pixel values of an image is clipping of highlights and underexposing shadow areas of the image. If d_L, d_a, and d_b represent a simple, global, linear shift of pixel values, it is possible for the pixel values to end up outside the range that can be represented by CIELAB space.[*] In such a case, the pixel must be clipped to the maximum or minimum representable value. This results in distinctly unpleasant and unnatural images. One possible solution is "soft" clipping, whereby the value of a pixel is clipped to some value close to the maximum representable value depending on how far it lies outside the representable range. We did not find the results of soft clipping satisfactory, and chose instead a somewhat unintuitive but remarkably effective method that adjusts the value of every pixel in the image by different amounts depending on the pixel value.

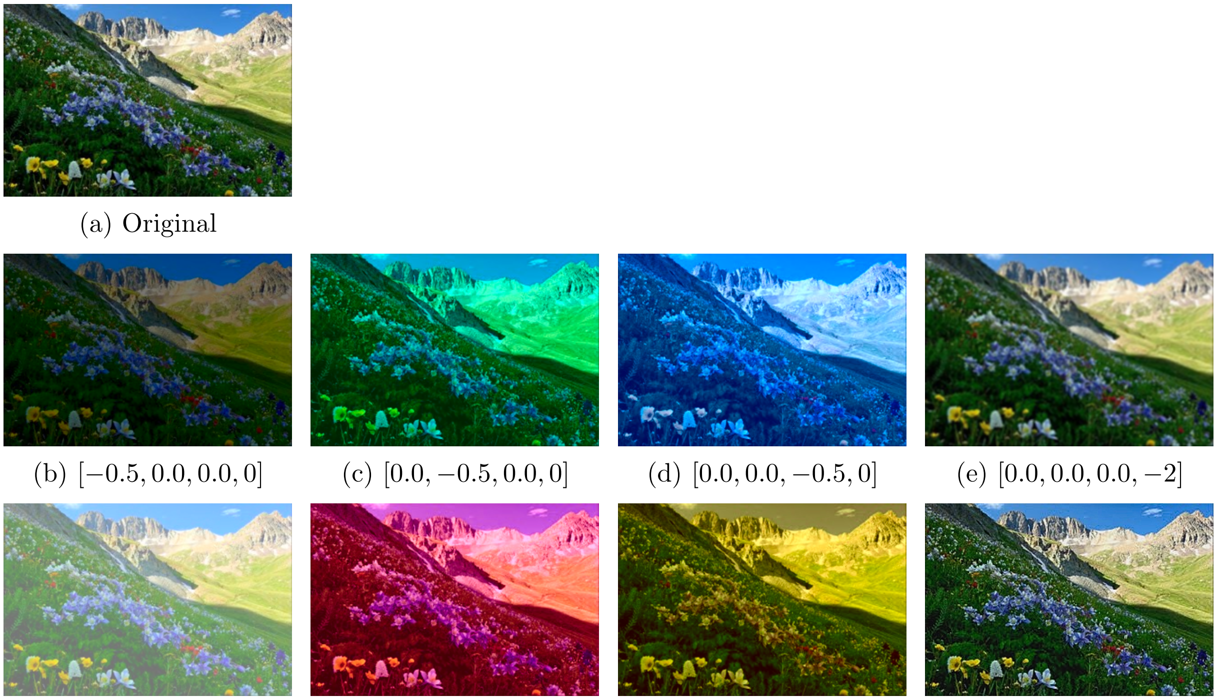
CIELAB space represents a finite range of values in the range $[L_{\min}, L_{\max}]$, $[a_{\min}, a_{\max}]$, and $[b_{\min}, b_{\max}]$. Assume that prior to adjustment each pixel in the image is represented by the set of values L_{old}, a_{old}, and b_{old}, and that we want to change those values to L_{new}, a_{new}, and b_{new}. Now consider the maximum difference between the new and old values if all the values are to remain within the representable range. This is simply the difference between the old value and the maximum or the minimum of the parameter, depending on the direction of the change. d_L, d_a, and d_b then represent the fraction of this difference for each image pixel, as follows:

$$L_{\text{new}} = L_{\text{old}} + (L_{\max} - L_{\text{old}}) * d_L \quad \text{if} d_L > 0 \tag{1}$$

$$L_{\text{new}} = L_{\text{old}} + (L_{\text{old}} - L_{\min}) * d_L \quad \text{if} d_L < 0 \tag{2}$$

where $d_L \in [-1.0, 1.0]$. The equations for the a and b values are defined similarly; d_a and d_b have the same range as d_L. It is easy to see that the value of each pixel is shifted by a different amount. Note that this transformation always results in lower contrast than the original image. To preserve a reasonable contrast for the modified images and to avoid extreme cases (all white or all black image), we restricted the range of the parameters to the interval

[*] More importantly, the pixel values must be in the RGB space of the device.



(a) Original

(b) $[-0.5, 0.0, 0.0, 0]$ (c) $[0.0, -0.5, 0.0, 0]$ (d) $[0.0, 0.0, -0.5, 0]$ (e) $[0.0, 0.0, 0.0, -2]$

(f) $[+0.5, 0.0, 0.0, 0]$ (g) $[0.0, +0.5, 0.0, 0]$ (h) $[0.0, 0.0, +0.5, 0]$ (i) $[0.0, 0.0, 0.0, +2]$

Figure 2. Image samples for data collection (parameter values: $[d_L, d_a, d_b, s]$)

$[-0.5, 0.5]$. Figure 2 demonstrates the range of parameter choices by showing the maximal difference along one of the dimensions, while the others are kept unchanged. The fourth parameter d_s represents sharpness and is in the range of $[-2, 2]$. Positive values represent a sharpening of the image and negative values represent a blurring of the image. Sharpening is done with an unsharp mask of radius 1.5, threshold 0, and mask strength equal to $100\% * s$. Blurring is done by convolution with a 3×3 Gaussian window, where d_s represents the standard deviation of the Gaussian. Again, maximal settings of the sharpness parameter are shown in Fig. 2, while the other parameters are kept unchanged.

2.2 Learning the Mapping

We used a *support vector regression (SVR)*[7] learner to learn a function mapping the four parameters onto user ratings. SVR provides learning on continuous input and output variables and requires a smaller number of training examples than other techniques, e.g., neural nets. It is important to keep the number of training samples fairly small because we need to train for each specific user and concept. Another strength of SVR is that, due to the use of a kernel function for mapping inputs into higher dimensional space, it can learn arbitrary nonlinear functions, provided the hyperparameters are tuned correctly. On the other hand, the inverse of such nonlinear functions is not easy to obtain but can be approximated using (hill climbing) optimization techniques.

As a benchmark, we also used a simple linear regression model to learn a mapping between a users ratings and the image parameters. For linear regression the mapping is determined individually for each parameter. Since there are four parameters, there are four mappings to learn.

It is expected that a model obtained by an SVR learner will be able to more accurately predict user ratings than a linear regression model. However, in order to generate a controller for the concept, an inverse of this model is necessary. This is because the controller must determine the parameters that will generate an image with a given rating. However, as we saw above, obtaining the inverse for an SVR model is not easy, while the inverse of a linear regression model is simple. We used a "hill climber" approach to approximate the inverse of the learned SVR model. The hill climber is a local search method that is not guaranteed to consistently result in the best approximation for the inverse model. There are some remedies for this, for example, one can run several hill climbers starting at different locations in the parameter space and select the best result. However, in our implementation, we used a simple single hill climber. Thus, our SVR implementation was not guaranteed to yield the best results.

2.3 User Study

In our experimental results, we collected data from six individuals.[†] Each user trained the system on three concepts: *light, crisp,* and *pleasant.* As we discussed in the introduction, the light and crisp concepts were chosen to see if the system would "properly" learn these concepts, given the fact that these simple concepts are more or less directly related to the selected low-level parameters (lightness to color luminance (L^*) and crispness to image sharpness s and color contrast, which in turn depends on color balance). Of course, as we have already pointed out, even for these simple concepts, the mapping is not trivial, and some users may associate these terms with other parameters as well. The pleasant concept was chosen to see if the system could correctly interpret the user ratings for what one would presume to be a more complex and subjective concept.

The example images for all three concepts and for all users were generated from the same original sample image, shown in Fig. 2, in order to minimize variation between subjects and concepts. This is a natural landscape photograph, chosen in an attempt to minimize the effect of the subject matter of the photograph on the user ratings. Each user rated 60 example images for each concept. 40 of these examples were unique, and the other 20 were repeats drawn from the base set of 40 examples. By comparing the users ratings of both instances of a repeated example it is possible to get an idea of how consistent the user is in his/her ratings and to examine how a user's consistency impacts the efficacy of the system. A user's "self-error" was calculated as the root mean square error between two vectors containing the user's first ratings of the set of 20 repeated images and the user's second ratings of the set, respectively.

2.4 Results

The system results were evaluated in two different ways. The first method of evaluation compared machine predicted ratings of examples to the human subject ratings of the same examples. The second method was subjective evaluation of system performance by the users.

For the first evaluation method we used 10-fold cross-validation to generate and test the machine generated ratings of the system. That is, for a set of 60 user-rated examples, the system was trained on 54 of these, and we used the learned model to generate ratings for the remaining six images. We then calculated the root mean square error between the human and machine ratings. This is one fold. The process was run nine more times with a different set of six test examples each time, so that each image in the set of 60 was used once for testing and nine times for training. Figure 3 shows box plots for the cross-validated error of the models learned for each concept. As expected, the SVR learner does outperform the linear regression learner. The difference is relatively small for the concepts light and crisp, and is larger for the concept pleasant. This is to be expected since pleasant depends on all four parameters and is inherently a more complicated, vague, and subjective concept than either light or crisp.

In the second evaluation method, each subject was asked to rate the performance of each of the controllers. That is, the subject selected different settings for the high-level concept and an image was displayed for each of the corresponding low-level parameters determined by the controller. The user then used a slider to indicate her/his satisfaction with the controller. Table 1 shows the mean subject satisfaction ratings of generated concept controllers, normalized to the $[0, 1]$ range (1 is best), and RMS self-errors, normalized to the $[0, 1]$ range and averaged over all subjects. The results are for the most part consistent with the cross-validation findings. The only exception is for the light concept, for which linear regression slightly outperforms SVR. However, as we have already shown, humans are not always consistent in their ratings, which, coupled with the small number of subjects, could account for this.

It is interesting to note the similarity between the subjects self-error and the error of the predictions of the SVR model. This similarity makes intuitive sense. An SVR learner is only capable of learning based on the data it is given. As such, it cannot be expected to predict ratings with any more consistency than that displayed by

[†] A full-scale user study for this system is beyond the scope of this paper.

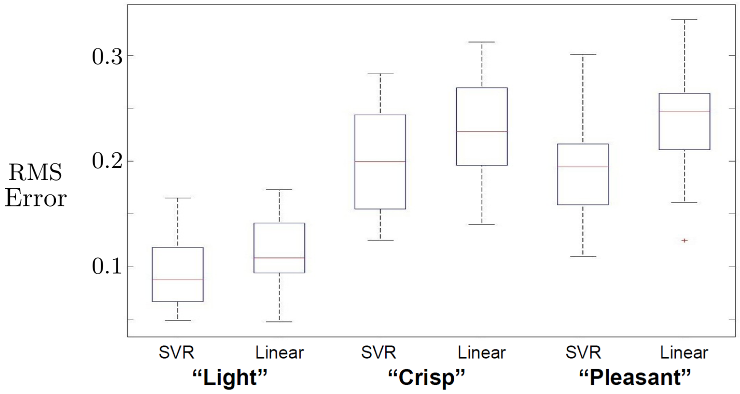


Figure 3. 10-fold cross-validated root mean square error of SVR and Linear Regression models for three image concepts (normalized to $[0, 1]$ range).

Concept	Mean Subject Satisfaction Rating		Mean RMS Self-Error
	Linear	SVR	
"Light"	0.88	0.83	0.11
"Crisp"	0.73	0.79	0.21
"Pleasant"	0.63	0.77	0.16

Table 1. Mean subject satisfaction ratings (range $[0, 1]$) of generated concept controllers and normalized RMS self-error.

the subject. We can infer that the subjects' self-error gives a rough bound on the performance of the learner. Given this bound, the learner has performed admirably, achieving results quite close to the bound in all cases. It should also be noted that the RMS error between two uniform random sets in the range $[0, 1]$ is on average $1/\sqrt{6} = 0.408$. Thus, the predictions made by the SVR model for the three concepts are on average slightly better than a two-fold improvement over random chance. Given the small training set for one concept and user (60 examples), this is a reasonable improvement. Figure 4 shows a typical example of how each system would respond to a user inquiry. Note that the linear regression controller fails at the most pleasant setting, while the SVR does not.

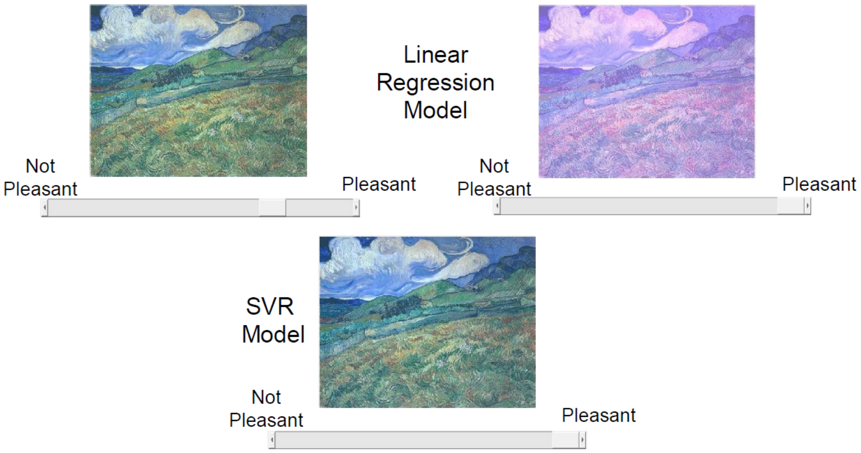


Figure 4. System example: Linear regression vs. SVR learning

3. MEASURING THE PERCEIVED ROUGHNESS OF VISUAL TEXTURES

While the main goal of this paper is the design of adaptive user interfaces that learn personalized mappings from image concepts to image parameters, with the end goal of providing a user with a personalized controller, the data collection and learning part of the system can be useful in their own right. Thus, an interesting application of the proposed learning techniques is in the study of human perception of the roughness of visual textures, a challenging problem that has received relatively little attention in the literature (e.g., in Refs. 6, 8–10).

One of the difficulties of conducting subjective tests to determine the perceived texture roughness is that differences in perceived roughness may be due to several factors. In an attempt to isolate one of those factors, subband variance, van Egmond *et al.*[6] conducted their subjective tests with both original and synthesized textures, the latter with uniformly distributed white noise of the same variance in each subband as that of the original texture. They kept the subband variances the same in order to create textures with statistics similar to the real textures, and also, to check the hypothesis that subjective roughness depends only on the subband variance. They also proposed a simple objective predictor of subjective visual roughness based on the variance of the coefficients of a subband decomposition.

This idea can be carried further in the context of the proposed system, which can explore the dependence of a high-level concept (roughness) to low-level parameters (subband variances). However, in addition to using machine learning, our study differs from that in Ref. 6 in two respects: (1) we scale the variances of the original textures to better populate the parameter space and to facilitate the learning, and (2) we use a steerable filter decomposition, which generates more realistic-looking textures.

3.1 Generating Samples

As discussed in Section 2, the choice of parameters is critical for an adaptive system to work. Based on the results of Ref. 6, we selected the variance of the coefficients in each subband as the low-level parameters that affect the roughness of a visual texture. However, in this study, we used the steerable filter decomposition[11] instead of the generalized quadrature mirror filter (GQMF) bank used in Ref. 6 for the texture analysis and synthesis. This is because the GQMF decomposition does not separate 45^o from -45^o orientation, resulting in inferior synthesis for textures with diagonal structure. On the other hand, in the steerable filter decomposition there is considerable overlap between subbands, which as we will see below, causes problems in texture synthesis.

As in Ref. 6, we used textures from the "CUReT" database,[5] cropped and converted to grayscale. We used four textures, samples 2, 14, 16, and 22, which correspond to polyester, roof shingle, cork, and lamb's wool, respectively. These textures are shown in the first row of Fig. 5. The selected set corresponds to lightning and viewing condition 122.[5] It is interesting to examine the distribution of subband coefficients of these textures. Figure 6 shows the histograms of the subband coefficients of "CUReT" Sample 2. Note that the distribution is approximately Gaussian in all subbands except the residual lowpass band. The distribution grows noisier at lower (subsampled) spatial frequencies, but may still be roughly approximated by a Gaussian.

To synthesize the texture samples for the subjective test, we used a variation of the texture synthesis method proposed by Heeger and Bergen.[12] In the Heeger-Bergen algorithm, a texture is synthesized by creating a Gaussian white noise image of the same mean as the original, computing the steerable filter decomposition, matching the histogram of each subband to the corresponding histogram of the original texture being synthesized, and recombining the subbands. The histograms are then matched in the original image domain. This procedure is iterated a few times to get a better match.

Given that the coefficient distribution of each subband may be approximated by a Gaussian, it is reasonable to assume that it is Gaussian (which is completely characterized by the mean and variance) and attempt to synthesize a texture by matching the variance of the distribution of each subband. (With the exception of the lowpass, the means are zero.) The procedure otherwise remains the same as that proposed in Ref. 12. The second row of Fig. 5 shows synthesis results with the same variances as the original CUReT textures shown in the first row, while the third and fourth rows show synthesis results with modified (scaled) variances. The figure clearly demonstrates that stochastic textures may be effectively synthesized by matching only the subband variances and the luminance distribution of the image. For a 4 scale and 4 orientation decomposition this leaves us with 18 variance values, and thus 18 parameters for an adaptive user interface.

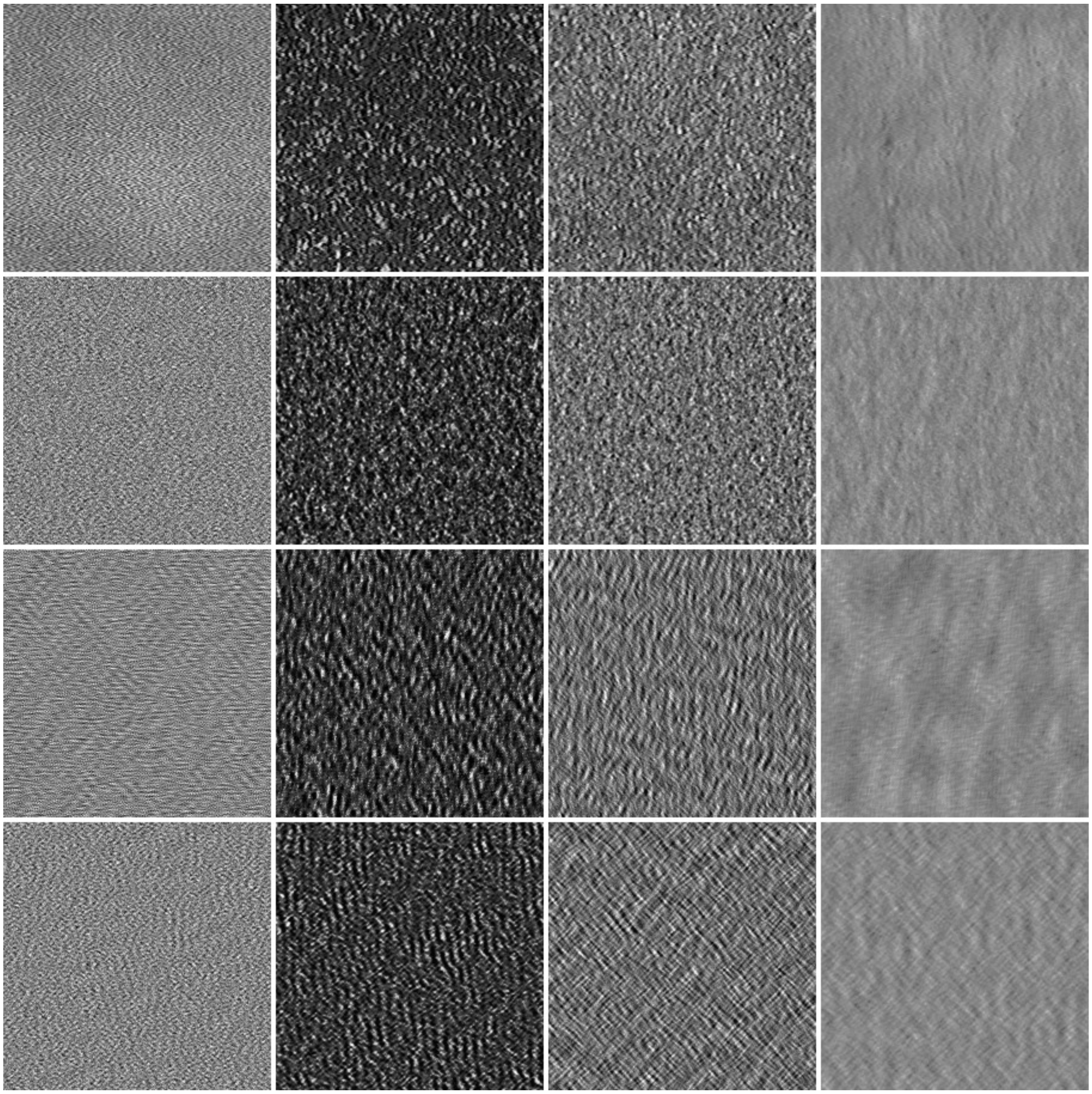


Figure 5. Original “CUReT” textures (top row); synthesized with approximately the same subband variances (second row); synthesized with modified variances (bottom two rows)

To generate the samples for a particular user, we select an original CUReT texture, compute the variances of the steerable filter decomposition subbands, multiply each variance by a scale factor, and synthesize with the procedure described above. The scale factor is chosen independently for each subband from a uniform distribution in the interval $[0, 2]$. We should point out, however, that the iterative procedure does not always produce the target variances. This can be attributed to the fact that in the steerable filter decomposition there is considerable overlap between subbands, which results in energy “leakage” between subbands. Accordingly, the system parameters are the resulting “final” variances, not the target scaled variances.

3.2 User Study

As we saw above, the concept of interest is roughness and the low-level parameters are the final subband variances. (From now on, we will refer to final variances simply as variances.) To learn a mapping from perception of roughness to the variance of each subband of a texture, we used SVR with a Gaussian RBF kernel and “sparse”

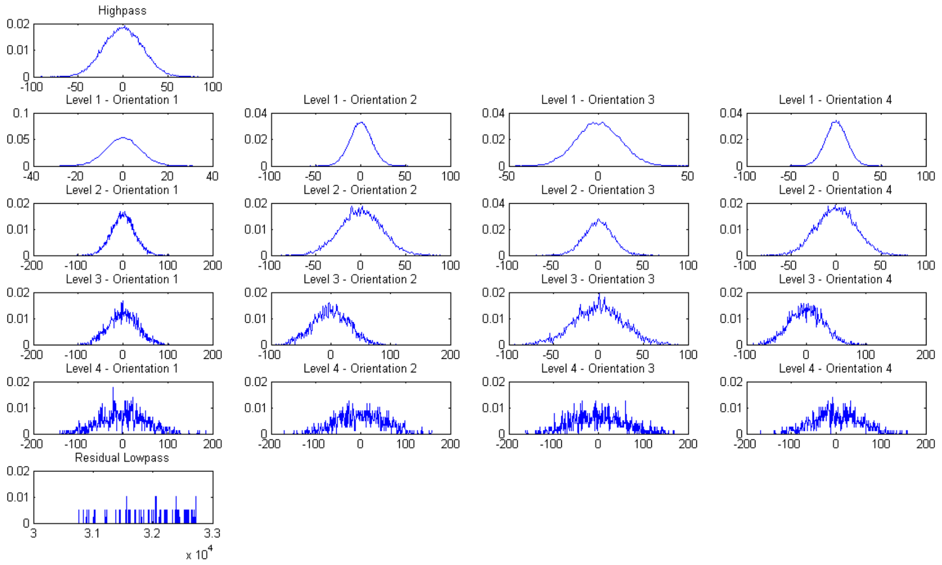
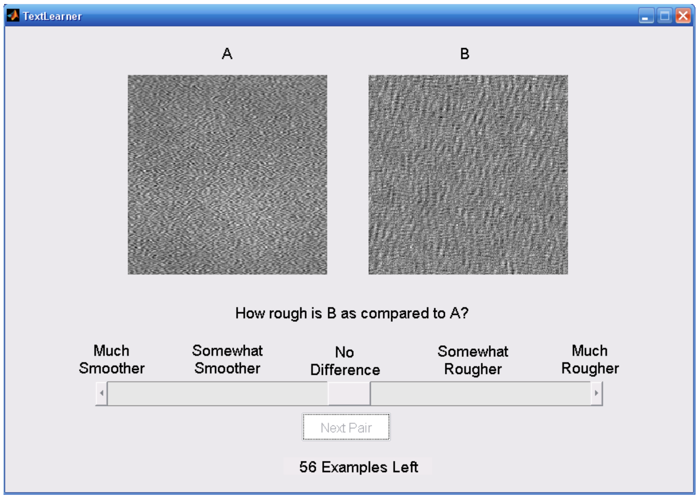


Figure 6. Subband Coefficients of CURET texture 2. First row: Residual highpass. Second to fifth rows: Levels (in decreasing radial frequency order) by row, orientations by column. Bottom row: Residual lowpass.



How rough is B as compared to A?

56 Examples Left

Figure 7. Subjective test for texture roughness experiment. Original texture was always on the left, synthetic on the right.

optimizer.[7] The subjective study included 12 subjects. Each subject was shown 60 modified examples of each of the four CUReT textures shown above, and rated how rough they were compared to the original texture. Out of the 60 examples, only 40 were distinct; 20 were used twice, in random sequence, to test subject self-consistency. The original and corresponding synthesized texture (synthesized from the original after scaling the subband variances) were presented side by side, with the original always on the left, as illustrated in Fig. 7.

The accuracy of the learned mappings was tested as described in Section 2. For each set of 60 examples (one user and one original texture) we used 10-fold cross-validation. The RMS error (normalized to $[0, 1]$ range) from one fold represents one point on the plots shown in the first four columns of Fig. 8. As we saw in Section 2.4, the RMS error between two uniform random sets in the same range is 0.408. Thus, according to the results of Fig. 8, the system performs about two and a half times better than random chance for each of the four textures. It is important to note, however, that the system was trained and tested for one subject and one image.

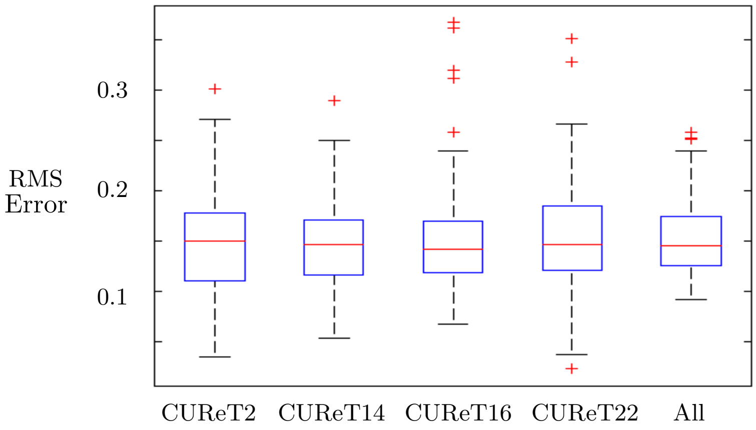


Figure 8. Cross-validation performed separately on each of the four CUReT textures and across all four textures.

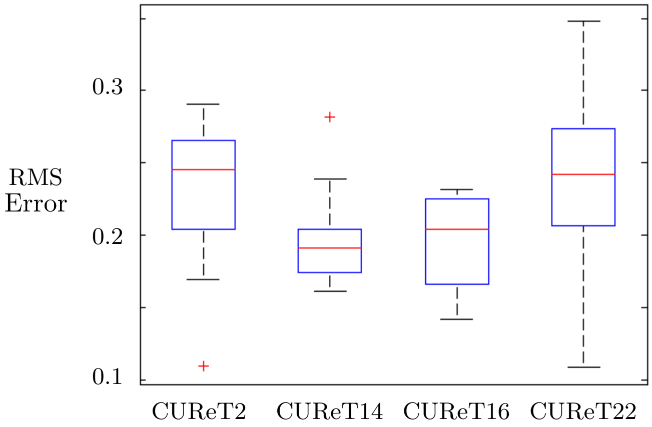


Figure 9. Cross-validation performed across all users for each texture.

It is interesting to test the ability of SVR to learn how a single subject perceives roughness across several textures. For this, we performed 10-fold cross-validation across all 240 examples (60 for each texture) shown to each user. Each fold consisted of 54 examples from each texture (for a total of 216 training examples) and 6 examples from each texture (for a total of 24 testing examples). That is, the samples from the different textures were interleaved, in order to ensure the robustness of the procedure. One fold is represented by one point in the fifth column of Fig. 8. Observe that the results for the combined learning are as good as those for the individual textures, perhaps even a little better (smaller variance and median, fewer outliers). Apparently, the potential disadvantage of training and testing over multiple images has been overridden by the four-fold increase in training data. In addition to the robustness of the proposed approach, this shows the system potential to improve with more training. In contrast to the real-time constraints for customized user interfaces, increasing the number of training examples does not cause any significant problems in this case.

We also tested the ability of the system to generalize among subjects by performing 10-fold cross-validation across all subjects, separately for each texture. Each fold therefore consisted of 648 training examples and 72 testing examples, again, interleaved among users. One fold is represented by one point on the plot in Fig. 9. In this case, the generalization results in performance loss, due to variations in user preferences.

Finally, we recorded each user's self-error for each set of 60 examples that correspond to one original texture. Each of these is represented by one point in the plot of Fig. 10. As we found out for the concepts presented in

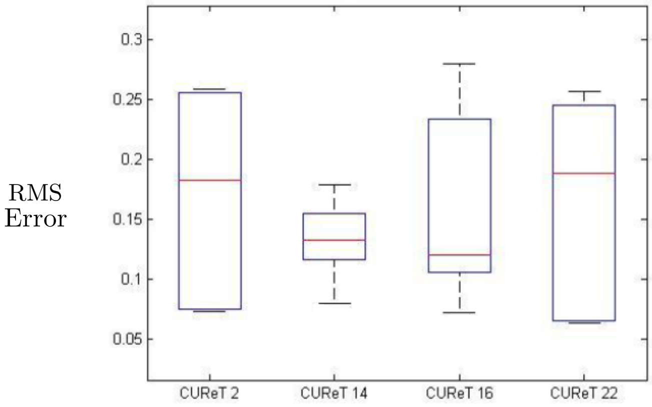


Figure 10. Subject average RMS self-error

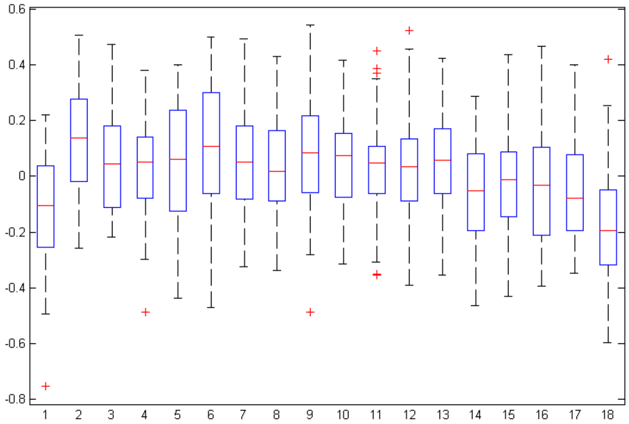


Figure 11. Correlation of (final) subband variances with subjective results

Section 2, the accuracy is approximately in line with the subjects' self-error, which demonstrates that the results are reliable. The low-level parameters for the SVR learner were the 18 subband variances. Figure 11 illustrates the correlation of each of those 18 parameters with the subjective roughness rating (one point of one box plot corresponds to the correlation between the subjective rating and subband variance of the corresponding subband for 60 examples generated for one subject from one original texture). It is clearly seen that no parameter has a significant correlation with the learned concept as all of the correlations more or less fall in the interval $[-0.2, 0.2]$. This shows that learning of subjective roughness based on subband variances would have been difficult if we had used a simple learner (such as independent linear regression used in Section 2) as no parameter by itself is highly correlated with subjective rating. However, the SVR learner could learn the concept based on those 18 parameters due to its non-linearity and high-dimensional kernel. Thus, even though no single subband variance has a significant effect on the roughness by itself, when all 18 subband variances are taken together, they can decide the roughness perception of texture. The very low correlation of subband variances makes it unlikely that any matrix that is based on linear combination of variances (such as proposed in Ref. 6) will produce accurate predictions on this data set. More comprehensive testing will be needed to explain the relationship of our results to those reported in Ref. 6.

4. CONCLUSIONS

We have developed an adaptive user interface that can learn the mapping between high-level concepts and low-level parameters. Moreover, it can accomplish that with relatively few user ratings, which makes it possible to implement in real time. We have shown that the proposed method outperforms linear regression and results in errors of the same order as the subjects' self-error, which provides a rough limit on learner performance.

One of the problems with the proposed approach is determining the inverse of the learned mapping, which is necessary for implementing the concept controller. We used a hill climbing method, which does not always produce consistent results. In future work we will seek a more accurate and efficient way to approximate the inverse of the model learned by SVR. This is a key step toward the ultimate goal of a system that can function in real time. We also plan to consider other parameters for image editing, such as image contrast, as well as other types of tone mapping.

We have also shown that the proposed techniques can be used to analyze the dependence of perceived texture roughness on low-level image parameters (subband variance). While the texture study is promising, more information is needed to properly specify a reliable control for roughness. We believe that more systematic user studies with more textures and better sampling of the parameter space will accomplish this goal.

ACKNOWLEDGMENTS

We want to thank Prof. Jack Tumblin and Arefin Huq for valuable discussions and insights. This work was supported in part by the National Science Foundation (NSF) under Grant Nos. IIS-0757544 and IIS-1049001. Any opinions, finding, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily relfect the views of NSF.

REFERENCES

[1] Sabin, A. and Pardo, B., "Rapid learning of subjective preference in equalization," in [*Audio Engineering Society Convention 125*], (Oct. 2008).

[2] Rafii, Z. and Pardo, B., "Learning to control a reverberator using subjective perceptual descriptors," in [*10th Int. Soc. Music Information Retrieval Conf.*], (Oct. 2009).

[3] Yang, C.-K. and Peng, L.-K., "Automatic mood transferring between color images," *IEEE Computer Graphics* **28**, 52–61 (Mar. 2008).

[4] Egmond, R. V., Lemmens, P., Pappas, T. N., and de Ridder, H., "Roughness in sound and vision," in [*Human Vision and Electronic Imaging XIV*], Rogowitz, B. E. and Pappas, T. N., eds., *Proc. SPIE* **7240**, 72400B–1–12 (Jan. 2009).

[5] Dana, K. J., van Ginneken, B., Nayar, S. K., and Koenderink, J. J., "Reflectance and texture of real-world surfaces," *ACM Transactions on Graphics* **18**, 1–34 (Jan. 1999).

[6] Egmond, R. V., Pappas, T. N., and de Ridder, H., "Subband analysis and synthesis of real-world textures for objective and subjective determination of roughness," in [*Human Vision and Electronic Imaging XV*], Rogowitz, B. E. and Pappas, T. N., eds., *Proc. SPIE* **7527** (Jan. 2010).

[7] Smola, A. J. and Schlkopf, B., "A tutorial on support vector regression," *Statistics and Computing* **14**, 199–222 (Aug. 2004).

[8] Ho, Y.-X., Landy, M. S., and Maloney, L. T., "How direction of illumination affects visually perceived surface roughness," *Journal of Vision* **6**, 634–648 (May 2006).

[9] Ho, Y.-X., Maloney, L. T., and Landy, M. S., "The effect of viewpoint on perceived visual roughness," *Journal of Vision* **1**, 1–16 (Jan. 2007).

[10] Bergmann Tiest, W. M. and Kappers, A. M. L., "Haptic and visual perception of roughness," *Acta Psychologica* **124**, 177–189 (2007).

[11] Simoncelli, E. P. and Freeman, W. T., "The steerable pyramid: A flexible architecture for multi-scale derivative computation," in [*Proc. ICIP-95, vol. III*], 444–447 (Oct. 1995).

[12] Heeger, D. J. and Bergen, J. R., "Pyramid-based texture analysis/synthesis," in [*Proc. Int. Conf. Image Processing (ICIP-95), vol. III*], 648–651 (Oct. 1995).