www.icgst.com

Extending the Scale Invariant Feature Transform Descriptor into the Color Domain

Robert H. Luke*, James M. Keller*, Jesus Chamorro-Martinez+

*Electrical and Computer Engineering, University of Missouri
Columbia, MO, USA, 65202
rhl3db@mizzou.edu, kellerj@missouri.edu,
+Computer Science and Artificial Intelligence, University of Granada
Granada, Spain
jesus@decsai.ugr.es

Abstract

In recent years, Lowe's Scale Invariant Feature Transform (SIFT) algorithm has become a widely used tool for object recognition. One shortcoming is that it only works on grayscale image intensities. In this paper, we propose extending the SIFT descriptor to also incorporate color information in a novel fashion. The keypoint descriptor proposed in this paper is a superset of the original feature vector and works exactly the same on grayscale or single band images as well as in regions of a color image that are essentially monochromatic. Examples are presented to demonstrate the properties and matching capability of the new descriptor in color image processing.

Keywords: *Color Computer Vision, Scale Invariant Feature Transform, Color Histogram, Object Recognition.*

1. Introduction

The Scale Invariant Feature Transform (SIFT) [1] has become a popular object recognition tool for Robotics and Computer Vision, [2,3,4]. Its popularity is due to its high matching accuracy even with moderate changes to scale and viewing angle. SIFT's robustness can be mainly attributed to its choice of feature descriptor. The descriptor is built using image gradients from gray scale intensities. The gradients represent the texture of the object and its geometric shape.

By focusing only on gray scale intensities, some unique objects can appear very similar. Figure 1 shows one such example. Color information is a descriptive feature and, in some cases, is necessary to differentiate objects. In [5], Swain reports on a descriptor based on color histograms. A set of histograms, one for each pixel of a sub window, is built to describe a training object in an image. This feature vector was highly accurate, but did not easily handle changes in scale and orientation. A more recent color descriptor [6, 7] was developed by Abdel-Hakim as an extension to the SIFT algorithm. Abdel-Hakim's feature vector indeed shows a significant improvement over the original descriptor but uses only a linear transform of red, green and blue (RGB) values, which intrinsically hold both color and intensity. We believe separating the color and gray scale information is important for a robust descriptor.

Another method that fuses color information with the SIFT descriptor is defined by Ancuti [8]. Here, color co-occurrence histograms (CCH) as well as gradient information from the original SIFT descriptor are used. The weakness with that method is that matching is performed using two separate measures for the original descriptor and the new CCH descriptor. In this form, the color information is used only to add matches not detected strictly from the gradient information. A more robust method might concatenate the gradient and color information so as to have a single distance metric and matching criterion.

Figure 1. Three objects with the same texture but unique colors. SIFT cannot differentiate between these objects because its descriptor is based on gray scale intensities.

Several other researchers have made extensions to the SIFT algorithm. PCA-SIFT [9] performs a principle component analysis of the descriptors to project into a lower dimensional space. Reference [10] selects keypoints from images using attention models so as to reduce the number of keypoints, minimize processing times, and increase matching accuracy. Zhang et al. [11] have also increased the speed of SIFT matching through CPU parallelization.

We propose combining hue and saturation with gradient information to define a more robust descriptor. We strived to logically extend the SIFT descriptor instead of using color as a secondary test for matching. Therefore, those readers familiar with SIFT should easily comprehend the color descriptor defined in this paper. A second advantage to this descriptor is the reuse of Lowe's matching criteria.

The remainder of the paper is organized as follows. Section 2 describes the original SIFT algorithm. Section 3 proposes an extension of the SIFT descriptor into the color domain. Section 4 displays test results comparing the old SIFT descriptor and the one defined in this paper. Section 5 concludes the paper.

2. SIFT

The SIFT algorithm uses Scale Space Theory [12,13] to find interesting locations in images called keypoints. To do this, a training image is incrementally blurred using a Gaussian kernel to create a stack of blurred images called an octave. The difference between each image in an octave is then computed. These are referred to as the Difference of Gaussian (DOG) images. Each pixel in each DOG image is checked against its eight immediate neighbors and the nine immediate neighbors in the scale above and below its scale. If the pixel is greater than or less than all the 26 neighboring pixels, the location is determined to be an extremum. The sub-pixel location of the extremum is determined using a second order Taylor expansion of neighboring pixel values. If the extremum lies upon an edge or in an area of low contrast it is disregarded. If the extremum passes this series of tests, it is likely to be the center of a unique area that will be found in future images, and is therefore considered a keypoint.

The next step is to find the keypoint's orientation. A single histogram of gradients with 36 bins is built using the gradients in the area around the keypoint location at the scale in which the keypoint was found. The bin with the largest value is defined as the keypoint's orientation. If another bin has a value greater than 80% of the maximum bin value, that bin's orientation is also used as a major orientation for this keypoint and will be given a unique descriptor. The area around the keypoint is rotated so that its major orientation points at the zero direction. This simplifies matching because only a single spatial orientation needs to be checked. The keypoint is then described using 16 histograms of gradients built from the grayscale intensity values for an area around the keypoint. Each of the 16 histograms has 8 bins, which results in a 128 dimensional feature vector. A database of keypoints can then be generated for a set of objects. Figure 2 shows the process of the SIFT algorithm finding and describing a single keypoint.

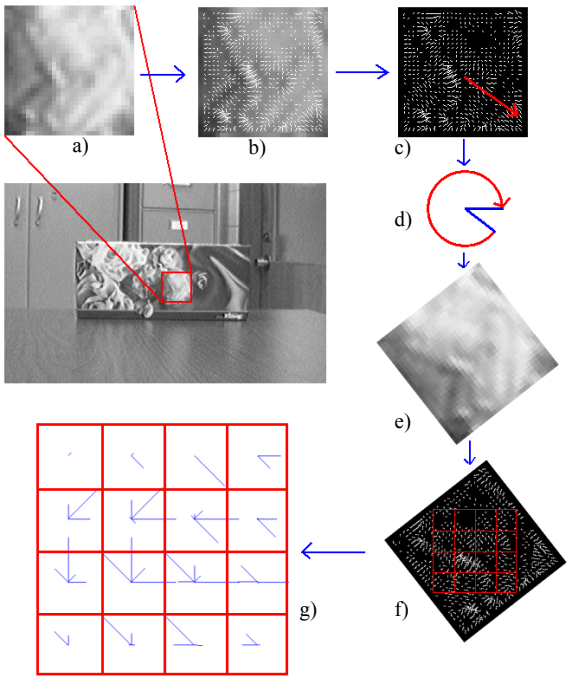


Figure 2. The sequence of finding and describing a single SIFT keypoint descriptor. a) First an extremum is found in scale space. If the extrema does not lie upon an edge or in an area of low contrast, it is considered a keypoint. b) The gradients in a sub window around the extremum. c) The gradients are used to find the major orientation of the keypoint. d,e,f) Each gradient is rotated by this amount to determine its final descriptor location and orientation. g) These gradients are then added to the appropriate histogram bins to form the final descriptor.

For matching, keypoints in a test image are found using the same process as was used for the training database. The Euclidean distance in feature space between each point in the test image and those in the known object database are determined. If for a given keypoint in the test image, the ratio of the shortest distance divided by the second shortest distance is less than some threshold value, (commonly 0.6-0.8), the associated keypoint pair is declared a match.

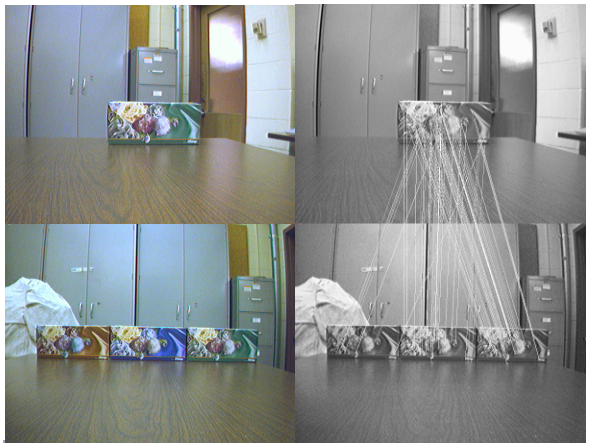


Figure 3. An example of incorrect matching using SIFT. Top Left: Training image. Lower Left: Test image. Right: Matching performed using the gray scale SIFT descriptor. The lines between images represent the detected corresponding keypoints. The green tissue box cannot be differentiated from the brown and blue ones using the gray scale SIFT descriptor.

The original SIFT descriptor is an extension of a descriptor developed by Edelman [14]. These descriptors are robust to changes in viewing angle, translation, and changes in scale. Edelman showed that matching was accurate with up to 20 degrees of rotation from the training image.

Edelman's descriptor has several useful attributes including high matching accuracy, ease of generation, and ease of matching (Euclidean distance). One main shortcoming of the descriptor is that it only uses gray scale information. This can lead to incorrect matching of objects that only differ in color. Figure 3 shows one such example.

3. Proposed Descriptor

Because the original SIFT descriptor so accurately describes geometry and texture, we decided to extend it with color information instead of creating an entirely new set of features. Needing only the color information, we extract the hue and saturation from the RGB input image. Several color spaces were considered for this descriptor. Some of the more well known spaces such as YCbCr and CIELAB accurately separate the color information from the RGB image, but the most elegant extension to the original SIFT descriptor came from the Hue, Saturation and Value (HSV) color space. The formulas used to transform from RGB to HSV space [15] are as follows,

$$\vee = Maximum, \tag{1}$$

$$\wedge = Minimum, \tag{2}$$

$$H \in [0,360), \tag{3}$$

$$S, V, R, G, B \in [0,1], \tag{4}$$

$$C_{\max} = R \vee G \vee B, \tag{5}$$

$$C_{\min} = R \wedge G \wedge B, \tag{6}$$

$$H = \begin{cases} 0, & if \quad C_{\max} = C_{\min} \\ 60\left(\dfrac{G-B}{B_{\max}-B_{\min}}\right), & if \quad C_{\max} = R \quad and \quad G \geq B \\ 60\left(\dfrac{G-B}{B_{\max}-B_{\min}}\right)+360, & if \quad C_{\max} = R \quad and \quad G < B \\ 60\left(\dfrac{B-R}{B_{\max}-B_{\min}}\right)+120, & if \quad G \\ 60\left(\dfrac{R-G}{B_{\max}-B_{\min}}\right)+240, & if \quad B \end{cases} \tag{7}$$

$$S = \begin{cases} 0 & if \quad C_{\max} = 0 \\ 1-\dfrac{C_{\min}}{C_{\max}} & else \end{cases} \tag{8}$$

$$V = \frac{(R+G+B)}{3}. \tag{9}$$

Hue is a circular value. This means that hues at 0 and 359 degrees are very similar shades of red, but if their similarity is computed using the difference in degrees, then they are interpreted as extremely different, which is incorrect. One solution is to use only the shortest distance between two hues in the circular space. Therefore, no two hues can be more than 180 degree apart. This is not a viable solution because such differences are relative. For example, the difference between red (0°) and green (120°) is 120°, the same as the difference between green (120°) and blue (240°). This results in ambiguity, which is undesirable. This approach and its ambiguities would further complicate a method that incorporated gradient information.

The main advantage of using gradients between gray scale intensities in the SIFT descriptor is that their magnitudes are relative. These relative values will remain nearly constant with changes in light intensity. On the other hand, color information (such as hue and saturation) are constructed so that they change little with changes in light intensity. This led us to a more elegant solution to a color feature vector. Instead of computing differences between pixels in HSV space, we use the hue and saturation of each pixel directly to construct a set of hue-saturation histograms describing local color.

As already stated, hue is a circular value ranging from 0 to 359 degrees. We extend the original SIFT features using another set of 16 histograms, each having 8 bins. Like the gradient direction in the SIFT descriptor's histogram of gradients, the hue value of a pixel can be used to determine the proper index in a color histogram. The saturation value of the pixel can then be added to the appropriate bins in the same manner as gradient magnitude in Lowe's original descriptor.

Because the hue and saturation can change slightly with light intensity, we interpolate the pixel values over neighboring histograms and bins similar to that of the SIFT intensity descriptor. The fraction of the saturation added to each bin is determined using linear interpolation with respect to spatial location and hue value. We next describe the equations used to add pixel saturation to histograms.

Histogram values are represented as, $h_{r,c,b}$, where r is the row location, c is the column location and b (if present) is the bin. The user determines the number of histograms and bins in the descriptor. We use a structure identical to the original SIFT descriptor. There are 4 rows of histograms ranging from 0, the top row, to 3, the bottom row. There are also 4 columns of histograms ranging from 0, the left most row, to 3, the right most row.

$$\begin{matrix} h_{0,0} & h_{0,1} & h_{0,2} & h_{0,3} \\ h_{1,0} & h_{1,1} & h_{1,2} & h_{1,3} \\ h_{2,0} & h_{2,1} & h_{2,2} & h_{2,3} \\ h_{3,0} & h_{3,1} & h_{3,2} & h_{3,3} \end{matrix}$$

Given a pixel p, taken from the sub window in a blurred image of scale space around a keypoint, the nearest histogram to the pixel p in the spatial domain is denoted

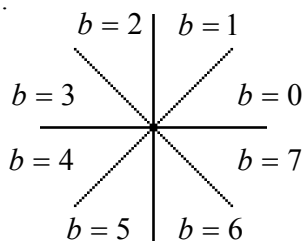
by, $h_{nearR,nearC}$. This corresponds to the nearest row and column histogram to the pixel. The nearest neighboring histograms are indexed with $neighR$ and $neighC$, which results in 4 histograms that are updated by the color values of a single pixel.

$$h_{nearR,nearC}$$
$$h_{neighR,nearC}$$
$$h_{nearR,neighC}$$
$$h_{neighR,neighC}$$

The bin index is used for further resolution. Similar to the row and column indexes, the bin indexes are for the nearest bin orientation to the pixels' orientation and to the nearest neighboring bin. Combining these indexes results in the 8 nearest bins to the current pixel location and orientation. Each pixel updates each of these eight bins.

$$h_{nearR,nearC,nearB}$$
$$h_{neighR,nearC,nearB}$$
$$h_{nearR,neighC,nearB}$$
$$h_{neighR,neighC,nearB}$$
$$h_{nearR,nearC,neighB}$$
$$h_{neighR,nearC,neighB}$$
$$h_{nearR,neighC,neighB}$$
$$h_{neighR,neighC,neighB}$$

There are 8 bins starting with b=0 for the bin nearest to zero degrees and increases counterclockwise to a maximum of 7.



If each side of a sub window has length l_w pixels, each side of a histogram's window is $l_h = \frac{l_w}{4}$.

The pixel's horizontal location is defined as p_x, vertical location as p_y and hue as p_h. The nearest histogram can be determined for a pixel as

$$nearR = \left\lfloor \frac{p_y}{l_h} \right\rfloor, \tag{10}$$

$$nearC = \left\lfloor \frac{p_x}{l_h} \right\rfloor. \tag{11}$$

Where
$\lfloor\ \rfloor$
denotes the floor function.

The neighboring histograms are defined by

$$\begin{aligned} &if \quad \frac{p_y}{l_h} - \left\lfloor \frac{p_y}{l_h} \right\rfloor < .5 \\ &\quad neighR = nearR - 1 \\ &else \\ &\quad neighR = nearR + 1 \end{aligned} \tag{12}$$

$$\begin{aligned} &if \quad \frac{p_x}{l_h} - \left\lfloor \frac{p_x}{l_h} \right\rfloor < .5 \\ &\quad neighC = nearC - 1 \\ &else \\ &\quad neighC = nearC + 1 \end{aligned} \tag{13}$$

Similar to spatial size, the hue bin angular length is

$$l_b = \frac{360^0}{8}. \tag{14}$$

The nearest hue bin is

$$nearB = \left\lfloor \frac{p_h}{s_b} \right\rfloor, \tag{15}$$

and the neighboring bin is found by

$$\begin{aligned} &if \quad \frac{p_h}{l_b} - \left\lfloor \frac{p_h}{l_b} \right\rfloor < .5 \\ &\quad neighB = \mathrm{mod}(nearB - 1, 8) \\ &else \\ &\quad neighB = \mathrm{mod}(nearB + 1, 8) \end{aligned} \tag{16}$$

With these values defined, a pixel's saturation value can be distributed via linear interpolation of to the 8 nearest hue histogram bins. The interpolation is performed using three weights, α, β and γ.

$$\alpha, \beta, \gamma \in [.5, 1] \tag{17}$$

$$\alpha = 1 - \left| \frac{p_y}{l_h} - \left\lfloor \frac{p_y}{l_h} \right\rfloor - .5 \right| \tag{18}$$

$$\gamma = 1 - \left| \frac{p_h}{l_b} - \left\lfloor \frac{p_h}{l_b} \right\rfloor - .5 \right| \tag{19}$$

The values of α and β are maximum for pixels located at the center of a histogram's location. These values

decrease linearly to .5 at the boundaries of the histogram. Similarly, γ is maximal for orientations in the center of a histogram bin and decrease to .5 at the boundaries.
With the interpolation variables, we can now update the values of the eight nearest histogram bins with the pixel's saturation p_s. All other bins in the descriptor are unchanged.

$$
\begin{aligned}
h_{nearR,nearC,nearB} &= h_{nearR,nearC,nearB} + p_s\alpha\beta\gamma \\
h_{neighR,nearC,nearB} &= h_{neighR,nearC,nearB} + p_s(1-\alpha)\beta\gamma \\
h_{nearR,neighC,nearB} &= h_{nearR,neighC,nearB} + p_s\alpha(1-\beta)\gamma \\
h_{neighR,neighC,nearB} &= h_{neighR,neighC,nearB} + p_s(1-\alpha)(1-\beta)\gamma \\
h_{nearR,nearC,neighB} &= h_{nearR,nearC,neighB} + p_s\alpha\beta(1-\gamma) \\
h_{neighR,nearC,neighB} &= h_{neighR,nearC,neighB} + p_s(1-\alpha)\beta(1-\gamma) \\
h_{nearR,neighC,neighB} &= h_{nearR,neighC,neighB} + p_s\alpha(1-\beta)(1-\gamma) \\
h_{neighR,neighC,neighB} &= h_{neighR,neighC,neighB} + p_s(1-\alpha)(1-\beta)(1-\gamma)
\end{aligned}
\tag{20}
$$

Figure 4 describes how the proper histograms and bins are determined for a given pixel, graphically representing the meanings of α, β and γ.

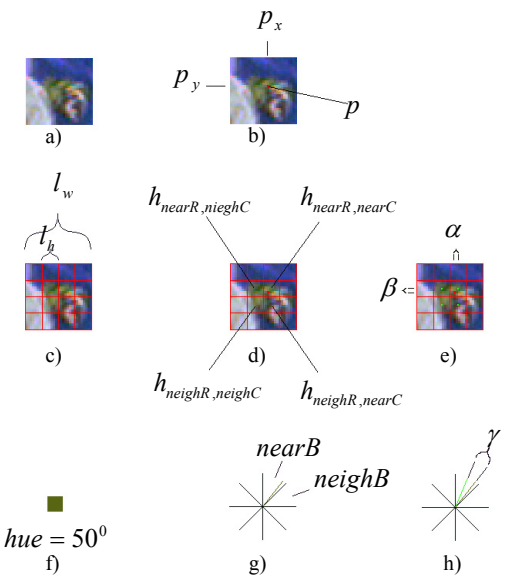


Figure 4. The nearest histograms and bins to a single pixel. a) A sub window of pixels around a keypoint. b) For the example in this figure, a single pixel p will be described. c) The histogram locations are placed over the subwindow. This shows the width of the subwindow, l_w, and each histogram, l_h. d) The nearest and neighboring histograms to the pixel p. e) The linear interpolation values of α and β. The closer a given pixel is to the nearest histogram, the higher the values of α and β. f) The hue of the pixel p is 50^0. g) The nearest and neighboring bin to the hue value of p. h) The linear interpolation value of γ.

For pixels near the border of the sub window, the nearest neighboring histogram will reside outside of the sub window. For these cases, only the nearest histogram is updated.

Figure 5 shows the entire process of placing a single color pixel into the appropriate color histogram bin.
The color descriptor is created using the same window around a keypoint as is used to create the histogram of gradients. The difference is that the color images are not blurred like the intensity images. Instead, the images are scaled in the same fashion as intensity images at the beginning of each new octave. Also, the structure of the color histograms is the same as the gray scale descriptor resulting in an additional 128 dimensional feature vector. The 128 values of the color vector are normalized. This normalization is also performed in the original SIFT algorithm to achieve robustness to changes in lighting. Like its gray scale counterpart, each value in the color feature vector has an upper threshold of .2, so that no single vector value is significantly greater than all others. Each of these values is then multiplied by the average saturation in the window. In this form, the color feature vector is a superset of Lowe's original descriptor. For example, if a grayscale image is used as input, the saturation values will be zero. Therefore, the color histogram descriptor will be a zero vector. Hence, the final output vector would be exactly the same as SIFT with a series of zeros attached. The matching process would therefore behave exactly like the original intensity descriptor matching.

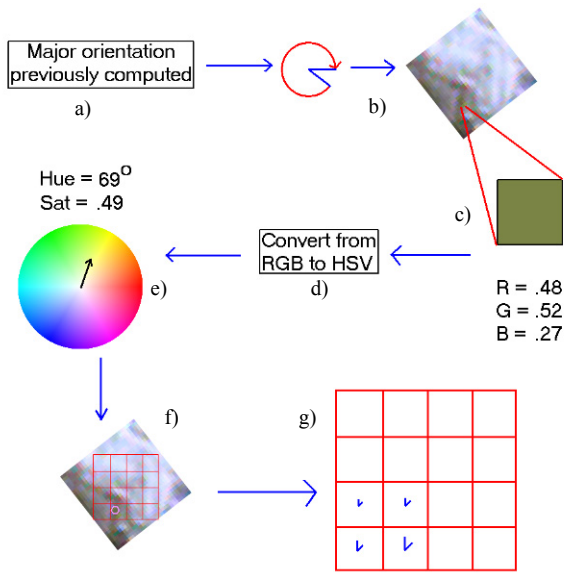


Figure 5. The process of adding a single pixel to the color descriptor. a) The major orientation was previously computed from gray scale gradients. b) The window around the keypoint is rotated to fix the major orientation at zero degrees. c) For this example, a single pixel is chosen to exemplify the addition of colors to the descriptor. d) The RGB values of the pixel are converted to hue and saturation. e) Hue and saturation of the pixel. f) The location of the pixel with relation to the 16 histograms. g) The saturation magnitude is added to the two nearest bins with relation to hue and the four nearest histograms with relation to spatial location.

Another useful attribute of this representation already touched upon is that it is robust to changes in lighting intensities. This is because the hue and saturation values will change only a small amount with lighting intensity variations. However, as with all color features, if the

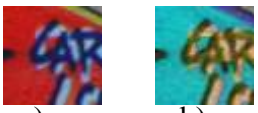
illumination is too low, the color information is unreliable.

It should be noted that most low cost cameras such as web cams will auto adjust the white balance so as to get the "best-looking" images. This change in white balance will result in an overall change in the image's colors, often making the image look too red, green or blue. If the change is too dramatic, the values of keypoint color descriptors will change synthetically and will result in a reduction of matches found. There are a couple of ways to fix this problem. First, a keypoint database could be built for a single object using training images from a variety of locations and lighting situations. This way, there will hopefully be a training setting similar to any possible test case. The second possibility would be to use a camera with programmable parameters. In that case, the white balance could be determined once and those settings could be used from then on.

4. Testing/Results

All testing was performed in Windows Vista 64 with 4GB of RAM and a 2.66GHz quad core Intel processor. All programs were written in C++ using Microsoft Visual Studio 2008 Express. Because this paper focuses on the features descriptor, all testing will focus on descriptor properties. Between 2.4 to 3.5 milliseconds are spent to create a 256 dimensional descriptor. In contrast, the original SIFT descriptor takes between 1.4-1.8 milliseconds to create each 128 dimensional descriptor. Overall processing time therefore is related to the complexity of the processed image, (number of keypoints), as opposed to the size of the image.

The properties of the color descriptor can be demonstrated by synthetically modifying the hue of a keypoint window. The subwindow around a sample keypoint is shown in figure 6 (a). This subwindow is then transformed into HSV space where the hue can be manipulated independently from the saturation and intensity. Figure 6 (b) shows the original subwindow with a hue change of 180^0. The subwindow hue is rotated by 1^0 to 360^0 resulting in 360 unique subwindows over which the original SIFT descriptor and our descriptor are built. Figure 7 shows the distance between the descriptors built from the modified subwindows and the descriptor from the original subwindow. As anticipated, the original SIFT descriptor shows nearly no change, being intensity oriented, while our descriptor changes drastically. Figure 7 also displays the results of performing this matching task on an unsaturated area of the image. Because the saturation is so low, our descriptor behaves much like the SIFT descriptor.



a) b)

Figure. 6 Modifying the hue of a keypoint subwindow. a) The original window. b) The window hue shifted by 180^0. Image from http://lear.inrialpes.fr/people/mikolajczyk/Database/

An unforeseen benefit of our feature vector is its robustness to changes in scale. Though SIFT claims to be scale invariant, the truth is that trained keypoints have an optimal scale range for matching. As scale increases or decreases, the descriptor for a given keypoint will also change. This is because SIFT features are based on gradients, which are built from neighboring pixel values. With scale change, image sub sampling changes neighboring pixel values which leads to a change in the overall descriptor. On the other hand, our color approach is based on the distribution of color over an area of pixels. Changes in scale will likely add, remove or smooth color values by only a small amount. Therefore, the overall representation changes minimally.

Figure 8 shows a keypoint window that was synthetically shrunk from 100%-20% in each dimension by 1% increments. The descriptor for the 60% window was used as the training subwindow. The distance to each of the 80 new feature vectors was then found. Using color alone, the descriptor changes only a small amount. On the other hand the computed distances stay very low. Of the total change to the concatenated descriptor defined in this paper, only around 30% of the change is due to the color features.

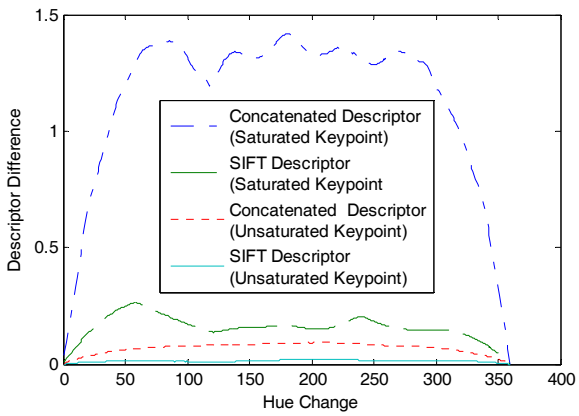


Figure. 7 Distance of hue modified subwindow descriptors from unmodified subwindow descriptor. Our color descriptor changes greatly with change in hue. In contrast, both descriptors perform similarly in unsaturated areas.

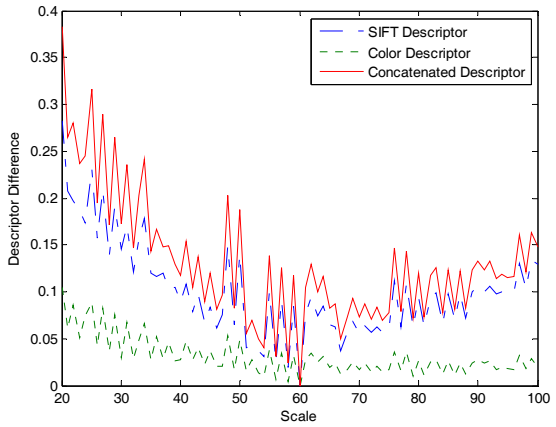


Figure. 8 Relation of synthetic change in scale to change in descriptor. The SIFT descriptor does not robustly handle changes in scale. The change in our color descriptor in minimal.

For object detection testing purposes, we decided to show the most extreme cases of SIFT failures. In the first case, we synthetically generate test images by swapping the red and blue channels of RGB training images. This is a simple change to the input data that results in approximately the same gray scale intensity, but a significant change in color. The second case, similar to the first, swaps the red and blue values in an image of a training object. The trained object is cut out of the swapped image and placed in a normal RGB test image. With this setup, the intensity of the trained object in the test image is nearly the same for both, but the colors are very different. In the third case, three different tissue boxes having the same pattern in varying colors are matched against each other. The final experiment uses outdoor images to test matching abilities from differing scales.

Figure 9 shows our first experiment using an original color image and the same image with the red and blue values swapped. The only colors that are unchanged by the swapping of the red and blue values are gray, green and purple hues. Therefore, only keypoints surrounded by these colors should match. But, because SIFT uses only gray scale intensities, the two images are nearly identical. This leads to a plethora of unwanted matches. Figure 9 (d) shows the same images matched using the new intensity and color histograms.

In the next experiment, shown in figure 10, we "cut and paste" the can with the red and blue values swapped into a test image with the can having the normal RGB values. Again, there should be no matches to the can with the red and blue values swapped. The SIFT gray scale descriptor produces many mismatches, while the color feature descriptor displays perfect matching.

Figure 11 shows a more realistic situation where the images are not modified. The use of color in the descriptor results in a significant reduction in mismatches and a significant number of correct matches. Here, there is only a single incorrect match utilizing our new color histogram features. The flowers on all the boxes have very similar color on all boxes and the single incorrect match can be attributed to this.

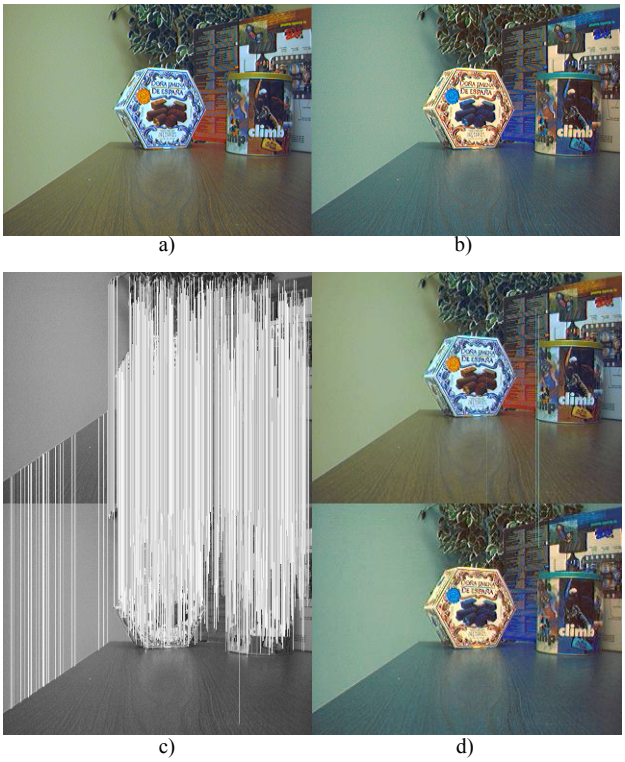


Figure 9. In this test, the red and blue values of an image were swapped. a) A training image. b) The same image as in a) but the red and blue channels are swapped. c) Matching performed using the SIFT gray scale descriptor. d) Matching performed with our new descriptor made up of gray scale and color information. The three matches found are in areas of grays or greens, areas nearly unchanged by a swapping of red and blue values.

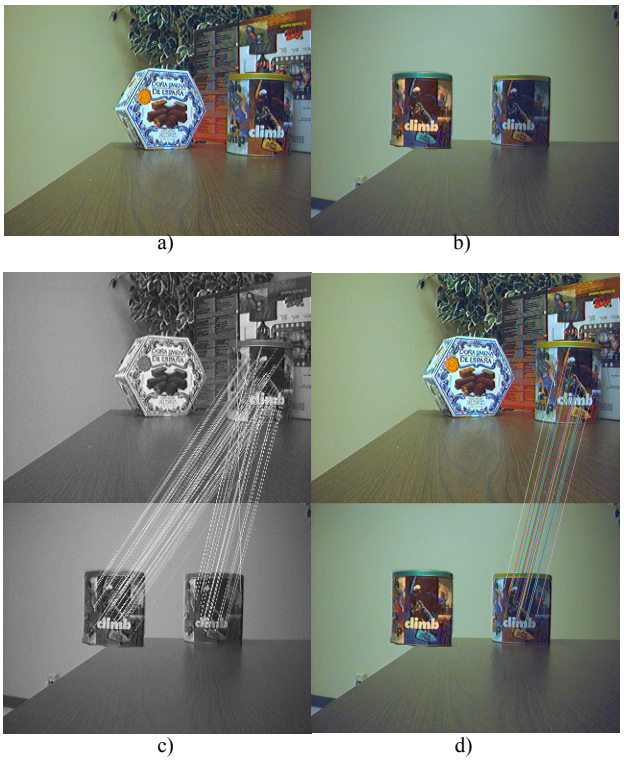


Figure 10. a) Training image. b) An image of the can has its values of red and blue swapped. This can is then seeded into another image with the original colors. c) The SIFT gray scale descriptor cannot differentiate between the two cans. d) Our descriptor using color as well as gray scale information matches to the proper can.

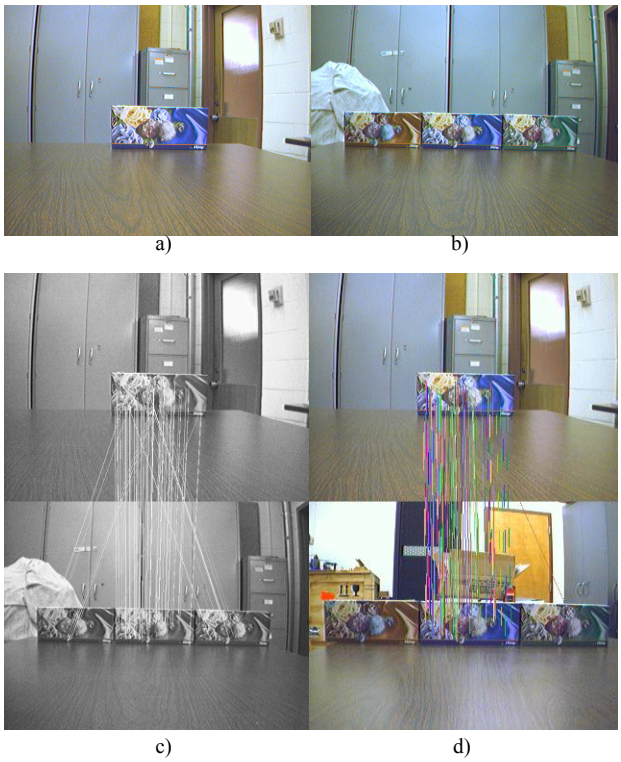


Figure. 11 This figure shows a more realistic situation where a blue tissue box is matched against itself as well as a brown and green box. a) A training image. b) A test image containing multiple tissue boxes with the same texture but unique colors. c) Sift has a large number of matches to the proper box, but also has a large number of mismatches as well. d) With our descriptor using color and gray scale, only a single incorrect match is found.

Possibly the most dramatic difference between the SIFT descriptor and that defined in this paper is shown using real changes in scale and rotation, images from the INRIA databasae [16]. Figure 12 shows the matching of two images that differ in scale and rotation. The SIFT descriptor produces 413 matches, whereas the concatenated descriptor results in 746 matches. Most of these new matches come from saturated areas in the trees.

5. Conclusion

This paper proposes a new color descriptor using hue and saturation appended to the original intensity histogram of SIFT. The saturation of each pixel's color is distributed to appropriate bins in histograms determined by hue. The structure of our color feature vector is designed to extend the representation of the SIFT algorithm having 16 histograms, with 8 bins per histogram. Our descriptor has the same format as that of SIFT, making it a logical extension.

The original SIFT approach only uses gray scale intensities and therefore color differences produces ambiguities. The color feature vector was designed to eliminate mismatches between objects of similar texture and geometry but differing color. Our experiments show that our color descriptor results in a comparable number of correct matches to the gray scale SIFT descriptor, but has significantly reduced incorrect matches. As well, the number of matches is significantly greater than SIFT with large changes in scale.

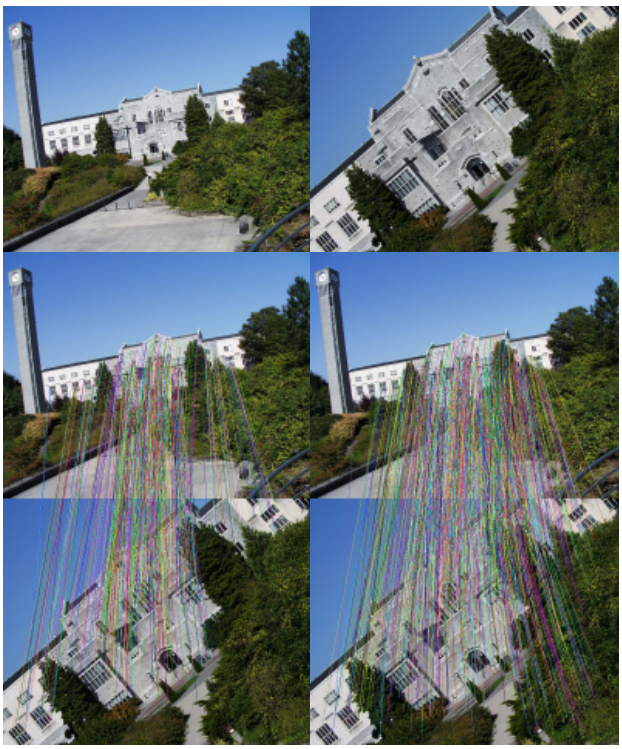


Figure. 12 Relation of change in scale and rotation to keypoint matching. Because the color descriptor (lower right) is much more robust to changes in scale than the SIFT descriptor (lower left), the number of matches nearly doubles. Image from http://lear.inrialpes.fr/people/mikolajczyk/Database/.

The use of color for computer vision is still uncommon. The lack of accurate representation of color as a set of features that uniquely identify objects is the main reason for color's infrequent use. Research such as ours and that of [6,7] and [8] have given new ways of representing color as a set of interesting features. There has recently been an increased interest in color image processing, which can only lead to better, more robust image analysis.

6. Acknowledgement

R. Luke is pre-doctoral biomedical informatics research fellow funded by the National Library of Medicine (T15 LM07089). This work is also supported by NSF under grant EIA-0325641.

7. References

[1] Lowe, D. 2004, Distinctive Image Features from Scale-Invariant Keypoints, I*nternational Journal of Computer Vision*, 60(2):91-110.

[2] Se, S., Lowe, D. and Little, J. 2002, Mobile robot localization and mapping with uncertainty using scale-invariant visual landmarks, *International Journal of Robotics Research,* 21(8):735-758.

[3] Luke, R.H., Blisard, S., Keller, J. and Skubic, M. 2005, Linguistic Spatial Relations of Three Dimensional Scenes Using SIFT Keypoints, *Proceedings of the IEEE International Workshop on Robot and Human Interactive Communication (RO-MAN)*, Nashville, TN pp. 704-709.

[4] Luke, R.H., Keller, J.M., Skubic, M. and Senger, S. 2005, Acquiring and Maintaining Abstract Landmark Chunks for Cognitive Robot Navigation, *Proceedings of the IEEE Intl. Conf. on Robots and Intelligent Systems (IROS)*, Edmonton, Alberta, Canada pp. 2566-2571.

[5] Swain M. J., and Ballard, D. H. 1991, Color Indexing, *International Journal of Computer Vision*, 7(1):11-32.

[6] Abdel-Hakim, A.E. and Farag, A.A. 2006, CSIFT: A SIFT Descriptor with Color Invariant Characteristics, *Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition* - Volume 2, pp. 1978-1983, 2006.

[7] Abdel-Hakim, A.E. and Farag, A.A., "Colored Local Invariant Features for Object Description", ICGST International Journal on Graphics, Vision and Image Processing, pp.73-80, Special Issue on Image Features and Analysis, 2006.

[8] Ancuti, C. Bekaert, P. SIFT-CCH: Increasing the SIFT distinctness by Color Co-occurrence Histograms, *5th International Symposium on Image and Signal Processing and Analysis*, 130-135, 2007.

[9] Ke Y., Sukthankar R. PCA-SIFT: A more distinctive representation for local image descriptors. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition,* pp. II-506-II-513, 2004.

[10] Gao K., Lin S., Zhang Y., Tang S., Ren H. Attention Model Based SIFT Keypoints Filtration for Image Retrieval, *Seventh IEEE/ACIS International Conference on Computer and Information Science*, pp. 191-196, 2008.

[11] Zhang G., Chen Y., Zhang Y., Xu Y. SIFT implementation and optimization for multi-core systems, *IEEE International Symposium on Parallel and Distributed Processing*, pp. 1-8, 2008.

[12] Witkin, A. P. Scale-space filtering, *Proceedings of the 8th International Joint Conferecne on Artificial Intelligence,* Karlsruhe, Germany, pp. 1019–1022, 1983.

[13] Koenderink, The structure of images, *Biological Cybernetics*, 50:363–370, Jan 1984.

[14] Edelman, S., Intrator, N. and Poggio, T. Complex cells and object recognition, *Unpublished manuscript*: http://kybele.psych.cornell.edu/edelman/archive.html, 1997.

[15] Gonzalez, R.C., Woods, R.E., *Digital Image Processing*, Prentice-Hall, Inc.: Upper Saddle River, New Jersey, 2002.

[16] Scale change images.

[17] http://lear.inrialpes.fr/people/mikolajczyk/Database/

Robert H. Luke received his B.S. in Computer Science in 2002 and a M.S. in Computer Science in 2005 from the University of Missouri, Columbia, MO, USA. He is currently a Ph.D. candidate in the Electrical and Computer Engineering Department at the University of Missouri.

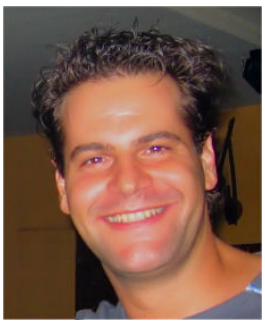
He is a National Library of Medicine Predoctoral Fellow. His research interests include computational intelligence, computer vision, and image processing.

Bob won the best student paper award at the IEEE International Conference on Fuzzy Systems (FUZZ-IEEE), WCCI 2008.

James M. Keller received the Ph.D. in Mathematics in 1978.

He holds the University of Missouri Curators' Professorship in the Electrical and Computer Engineering and Computer Science Departments on the Columbia campus. He is also the R. L. Tatum Professor in the College of Engineering. His research interests center on computational intelligence: fuzzy set theory and fuzzy logic, neural networks, and evolutionary computation with a focus on problems in computer vision, pattern recognition, and information fusion including bioinformatics, spatial reasoning in robotics, geospatial intelligence, sensor and information analysis in technology for eldercare, and landmine detection. His industrial and government funding sources include the Electronics and Space Corporation, Union Electric, Geo-Centers, National Science Foundation, the Administration on Aging, NASA/JSC, the Air Force Office of Scientific Research, the Army Research Office, the Office of Naval Research, the National Geospatial Intelligence Agency, and the Army Night Vision and Electronic Sensors Directorate. Professor Keller has coauthored over 250 technical publications.

Jim is a Fellow of the Institute of Electrical and Electronics Engineers (IEEE) for whom he has presented live and video tutorials on fuzzy logic in computer vision, is an International Fuzzy Systems Association (IFSA) Fellow, a national lecturer for the Association for Computing Machinery (ACM), an IEEE Computational Intelligence Society Distinguished Lecturer, and a past President of the North American Fuzzy Information Processing Society (NAFIPS). He received the 2007 Fuzzy Systems Pioneer Award from the IEEE Computational Intelligence Society. He finished a full six year term as Editor-in-Chief of the *IEEE Transactions on Fuzzy Systems*, is an Associate Editor of the *International Journal of Approximate Reasoning*, and is on the editorial board of *Pattern Analysis and Applications, Fuzzy Sets and Systems, International Journal of Fuzzy Systems*, and the *Journal of Intelligent and Fuzzy Systems*. He is currently the Vice President for Publications of the IEEE Computational Intelligence Society. He was the conference chair of the 1991 NAFIPS Workshop, program co-chair of the 1996 NAFIPS meeting, program co-chair of the 1997 IEEE International Conference on Neural Networks, and the program chair of the 1998 IEEE International Conference on Fuzzy Systems. He was the general chair for the 2003 IEEE International Conference on Fuzzy Systems.



Jesús Chamorro was born in 1972. He received the MS and PhD degrees, both in Computer Science, from the University of Granada in 1995 and 2001 respectively. Since 1996, he is member of the Computer Science Department at the University of Granada where he is now an Associate Professor. His research interests include motion analysis and optical flow estimation, information retrieval from image databases, soft computing image processing, and biomedical image analysis. His research work is summarized in over 50 papers in scientific journals and conference proceedings, and more that 10 research projects.