Improvement on Arty Shapes

Yuqi Shi 1003818700 yuqi.shi@mail.utoronto.ca
Jiaming Hu 1003820706 horis.hu@mail.utoronto.ca

1. Abstract:

This project implements and improves a machine learning application which converts photographs into abstract style artworks. Implementation part based on the paper Arty Shapes. [1] Four steps are followed to implement the functions in the original paper, including: Segment image using multiscale normalized cut (MN-cut) segmentation technique, Fit shapes into segmented part, Select best-fitted shapes automatically and Rendering work. Improvement work: our project improves the segmentation step in original paper. We firstly compare several state-of-art segmentation techniques and choose simple linear iterative clustering (SLIC) segmentation technique for its good performance in terms of under-segmentation error, boundary recall and time complexity. We apply both of MN-cut segmentation and SLIC segmentation into arty shape abstraction technique. The comparison images of each step are displayed in Section 4.

2. Introduction：

In recent years, producing abstract synthetic artworks becomes quite poplar based on Non-Photorealistic Rending techniques. Original abstraction techniques are mostly implemented using rather simple geometry primitives such as rectangles. The advantage of abstraction technique that Arty Shapes brings out is the invariance of fitting process and the possibility of fitting image with non-analytical geometrical figures such as triangles and convex hull. Furthermore, a novel shape classifier is provided which can automatically decide the 'best' shape representation of the segmented regions. This advances can simulate the works of geometric abstract art artist such as Kandinsky.

Arty Shapes abstraction technique can be implemented into four steps. Users firstly need to define the number of segmented regions N which is send as a input parameter into multiscale normalized cut segmentation method. Then we fit different geometry primitives such as rectangles, triangles and convex hull into segmented region. The next step is using a classifier to select the best-fitted shapes automatically for segmented region which increases usability. Rendering work creates some paper-cut effect to original image.

However, the segmentation method that applied in the first step of arty shapes technique has quite high time complexity. Multiscale normalized cut (MN-cut) segmentation method takes a rather long time to process which makes it quite inconvenient when applied in practical application. In this project, we replace the MN-cut algorithm with SLIC segmentation algorithm. Compared with MN-cut technique, SLIC provides more accurate adhering boundaries and shorter operating time.

3. Implementation & Improvement

3.1 Implementation of Arty Shapes

3.1.1 Segmentation

The segmentation algorithm applied in Arty Shapes is developed by Professor Timothee Cour called multiscale normalized cut in 2005. [2] It is chosen for the advantage of working

on many different scales of the images in parallel rather than repetition and allowing to segment images into many different scales to capture both fine and coarse level details. With an image I, a corresponding graph is created and represented by G= (V, E, W). V is the graph nodes in G corresponding to the pixels in image I. The nodes within certain distance Gr are connected by graph edges E. The graph connection radius Gr should be balanced between the segmentation effect and the computation cost. To decrease the time for computation, we need to cut down the number of iteration and speed up each iteration.

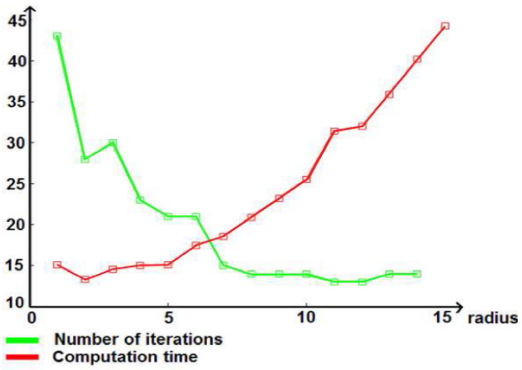

Figure 1. Effect of different graph connection radius

We measures the similarity of a pair of pixel which belongs to a same segmented region naming multiscale affinity matrix. NP hard is one of the problem every segmentation algorithm would meet when finding the optimal Ncut graph partitioning. A spectral image segmenting technique enables us to settle the NP problem using a continuous space solution rather than the discrete one by computing the K eigenvectors matching with K largest eigenvalues represented in the Equation 1 where D is a diagonal matrix represented in Equation 2.

$$WV = \lambda DV \quad \textbf{(1)}$$

$$D(i,i) = \sum_{j} W(i,j) \quad \textbf{(2)}$$

After calculation, we discretize V to X by normalizing the rows of V and then find the rotation R that brings it to the nearest binary indicator vector X which is called multiscale partitioning matrix. For linking all coarser levels to the most detailed graph scale, the coarse scale segmentation should be the mean value of all fine scale segmentations locally. Segmentation costs are defined to communicate across different graph scales. Cross scale constraint matrix is used to propagate segmentation cost information across the scales to reach a consonant segmentation at all scales. Constrained multiscale normalize cut algorithm is chosen as the segmentation criterion represented in Equation 3 supporting our segmentation work.

$$maximize\varepsilon(X) = \frac{1}{K}\sum_{l=1}^{K}\frac{X_l^T W X_l}{X_l^T D X_l} \quad \textbf{(3)}$$

The left Image in Figure 2 MN-cut segmentation result is the original image that we input into the system, after setting the number of segments to 20, we could achieve the result as the right image in Figure 2. The graph is segmented in a semantic way.

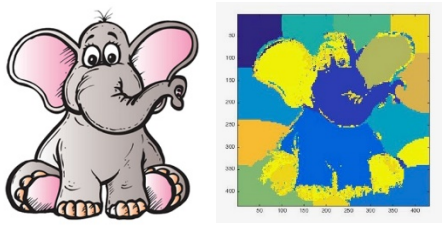

Figure 2. MN-cut segmentation result

3.1.2 Fit Shapes to Segmented Regions

The second step is shape fitting work. After segment the image, we are able to fit five different shapes to each region: circles, rectangle, triangles, superellipse and a 'robust' version of convex hull. We use the method of normalization for fitting circles, rectangle and triangles. [3] Least square method [4] is used to fit superellipse. A symmetric convexity measuring algorithm is

applied for fitting robust convex hull. [5]

Using super ellipse fitting work as an example, If the program is given a set of pixel list as the left part in Figure 3, least square method would create an output as the right part in Figure 3 which best represent the pixel list by a superellipse.

```
pixel_float
list: 0
303.541504 195.498184
291.584015 187.770752
279.235504 185.935776
265.887482 181.462997
246.087997 180.814728
236.649323 188.962921
231.980850 203.457520
219.476135 220.147659
202.402817 230.966690
191.143463 245.790863
174.271683 257.826782
154.197617 265.155609
132.349380 271.524353
116.801163 277.019928
102.755295 281.479675
88.018517 277.189758
71.943687 276.941010
64.891319 270.458466
58.862080 260.593933
55.974617 247.385651
-1 -1
```

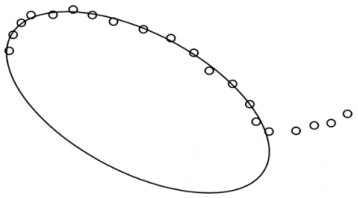

Figure 3. Ellipse Pixel List and Fitting Results

As for the problem of filling color to the shape region, a decision should be made between either use the mean color of all points on shape edge or the mean color of all points in segment region. The second method is taken for it considers more samples which may improve accuracy. Following two renderings are the result of filling all rectangles and the result of filling all circles in Figure 4. Fitting rectangles and circles.

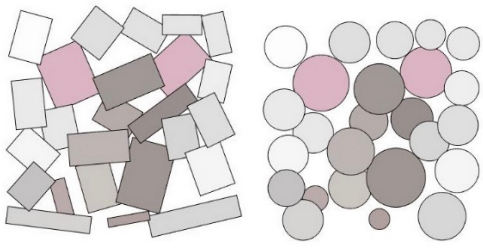

Figure 4. Fitting rectangles and circles

3.1.3 Select best-fitted shapes for segments automatically

The third and the most important part is automatic shape selection part. A decision tree algorithm -- C4.5 is used to classify the regions into different shapes. To build a decision tree, we need to perform average entropy calculations on the complete data set for each attribute and then select the one with lowest degree of entropy to build small decision trees which simplify our calculations. n_b is the number of instances in branch b, n_{bc} is the number of instances that belong to class c in branch b and the variable n_t is the total number of all instances in all branches. The average entropy is represented as E in equation 4.

$$E = \sum_b \frac{n_b}{n_t} \times \left[\sum_c -\left(\frac{n_{bc}}{n_b}\right) \log_2 \left(\frac{n_{bc}}{n_b}\right) \right] \quad (4)$$

This project selects some training images from Berkeley segmentation database and segment them into many different scales using the segment method mentioned above. It allows us to achieve many regions with a large variety of shapes. Each of the region is represented by a feature vector. The feature vector consists of the errors between each fitted shape models and the segments. We calculate the errors of the shortest distance to the fitted shape at each data point. The distribution of point errors including mean, standard deviation, skew and kurtosis are used to describe regions. A C4.5 decision tree is built using these training data, we can then give right class for the new objects. The data achieved from the second stage needs to be retransformed into the form that suitable for C4.5 training. The decision tree generated by c4.5 decision tree as follows in Figure 6 Decision Tree. The result after step three automatic shape selection is shown in Figure 5 Elephant Automatic Shapes Selection results.

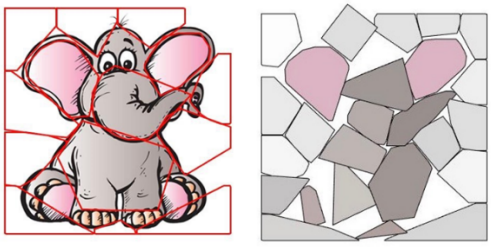

Figure 5. Elephant Automatic Shape Selection

```
Decision Tree:

supKurtosis <= -1.8429 :
|   rectKurtosis <= -0.96067 : circle (6.0/2.0)
|   rectKurtosis > -0.96067 : superellipse (10.0/1.0)
supKurtosis > -1.8429 :
|   rectKurtosis <= -1.2718 : rectangle (25.0/2.0)
|   rectKurtosis > -1.2718 :
|   |   rectSd <= 0.033323 :
|   |   |   rectSd <= 0.028309 : robustconvexhull (8.0/3.0)
|   |   |   rectSd > 0.028309 : rectangle (27.0/1.0)
|   |   rectSd > 0.033323 :
|   |   |   supSd <= 0.027773 : superellipse (9.0)
|   |   |   supSd > 0.027773 :
|   |   |   |   triSd <= 0.031108 : triangle (11.0)
|   |   |   |   triSd > 0.031108 :
|   |   |   |   |   circKurtosis > 1.7115 : triangle (10.0)
|   |   |   |   |   circKurtosis <= 1.7115 :
|   |   |   |   |   |   triKurtosis <= -1.7245 : triangle (8.0)
|   |   |   |   |   |   triKurtosis > -1.7245 :
|   |   |   |   |   |   |   rectMean <= 0.03785 : rectangle (10.0/1.0)
|   |   |   |   |   |   |   rectMean > 0.03785 :
|   |   |   |   |   |   |   |   circKurtosis <= -1.3336 : rectangle (10.0/2.0)
|   |   |   |   |   |   |   |   circKurtosis > -1.3336 :
|   |   |   |   |   |   |   |   |   supSkew <= -0.092613 :
|   |   |   |   |   |   |   |   |   |   circMean <= 0.14752 : robustconvexhull (3.0/1.0)
|   |   |   |   |   |   |   |   |   |   circMean > 0.14752 : rectangle (7.0)
|   |   |   |   |   |   |   |   |   supSkew > -0.092613 :
|   |   |   |   |   |   |   |   |   |   circSd <= 0.044915 : circle (6.0/2.0)
|   |   |   |   |   |   |   |   |   |   circSd > 0.044915 :
|   |   |   |   |   |   |   |   |   |   |   circMean <= 0.05733 :[S1]
|   |   |   |   |   |   |   |   |   |   |   circMean > 0.05733 :
|   |   |   |   |   |   |   |   |   |   |   |   supMean <= 0.030966 :[S2]
|   |   |   |   |   |   |   |   |   |   |   |   supMean > 0.030966 :
|   |   |   |   |   |   |   |   |   |   |   |   |   robustCHSkew <= 1.8777 :[S3]
|   |   |   |   |   |   |   |   |   |   |   |   |   robustCHSkew > 1.8777 :[S4]
```

Figure 6. C4.5 Decision Tree

3.1.4 Rendering

The last part of the core function is the rendering work to produce 'paper-cut' effect artwork. The project treats the layer with larger shapes as 'background' and the detailed shapes layer as 'foreground'. Shapes from the top layer are rendered whose colour deviates from that of the shape underneath above a certain threshold. We measure the color differences in just noticeable difference in LAB color space. For example, taking two colors (L1, a1, b1) and (L2, a2, b2) the color difference ΔE_{12} is defined as follows.

$$\Delta E_{12} = \frac{\sqrt{(L_1-L_2)^2+(a_1-a_2)^2+(b_1-b_2)^2}}{\text{jnd}} \quad \textbf{(5)}$$

where jnd is approximated as 2.3 in CIELAB color space. [6] By setting a threshold on ΔE, the degree of detail to save on the top layer can be controlled. Higher threshold leads to fewer shapes would be rendered and vice versa. Figure 7 is the result when we set E is 5.

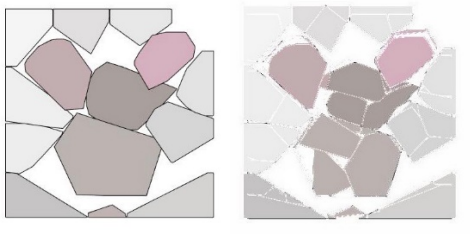

Figure 7. Rendering results of Elephant

3.2 Improvement on segmentation method

3.2.1 Related work

Arty Sharps used multiscale normalized cut algorithm to divide the original graph in the first step. Besides it, there also exist several state-of-art segmentation techniques can do similar work. In this part, we provide a summary of these available segmentation algorithms. In the following paper, N means the number of pixels of input image for segmentation algorithms.

Normalized cut algorithm (NC05) [2]: text cue and contour are used to partition all the pixels in a graph recursively. It can minimize the cost function of the adhering boundaries globally. Therefore, its super-pixels are quite regular. However, the running time is relative slow. Although it is improved by Greg Mori (NC10) [7]and the time complexity of Normalized cut is $O\left(N^{\frac{3}{2}}\right)$. It still can't work efficiently when process graph with large pixel numbers.

Turbopixel method (TP09) [7]: this algorithm uses geometric flow to expand set of seed locations, which segment graph by local image gradients. It divides graph into uniform size to get better adhering boundaries effect. And the time complexity of Turbopixel is O(N). However, during the test process, its performance is worse than SLIC in terms of boundary recall.

Simple Linear Iterative Clustering (SLIC) [8]: It is an iteration algorithm, which used k-means to generate super-pixels. K is the number of segments should be achieved in output. There exists five dimensions, l, a, b, for colors and x, y for the locations. Moreover, locations of seeds move to the lowest gradient position in a 3 × 3 neighborhood iteratively. The time complexity of SLIC is O(N).

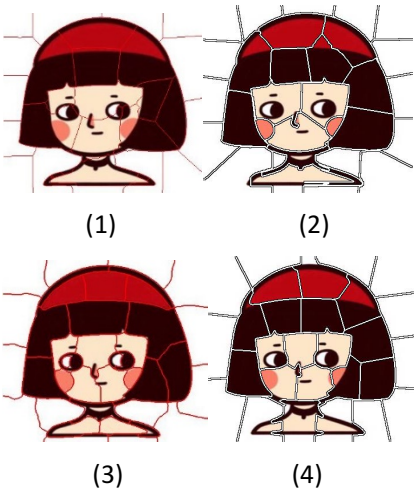

Figure 8. Four picture examples segmented into 20 parts. (1)TP09 (2) SLIC (3) NC10 (4) NC05

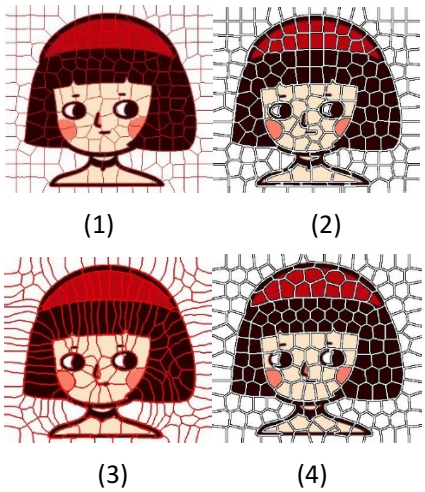

Figure 9. Four picture examples segmented into 200 parts. (1)TP09 (2)SLIC (3) NC10 (4) NC05

3.2.2 Algorithms performance evaluation

A comparison among these four super pixel algorithms is made to evaluate the performances in terms of time complexity and adhering boundary effect [7] [10]. To test the performance, the Berkeley dataset [9] is used in this paper. Berkeley dataset contains 300 pictures with 481x321 pixels and their corresponding ground true boundaries. 200 pictures are for training and 100 pictures are for testing.

3.2.2.1 Time Complexity

In order to apply Arty shape in mobile application or website, the running efficiency is important for the segmentation algorithms. In this paper, a comparison made among current algorithms, including NC05, NC10, TP09 and SLIC. The comparing result is presented in Figure 10, 11, 12. The result is achieved by compiling an increasing number of pixels, on Intel® Core™ i7-4700HQ CPU @ 2.40GHz × 8 processor with 8GB RAM. As shown in the figure, NC05 has the slowest running time, followed by TP09 which is 7 times faster than NC05 and NC10 has a similar efficiency as TP09. SLIC servers the best time complexity, which is nearly 70 times faster than NC05.

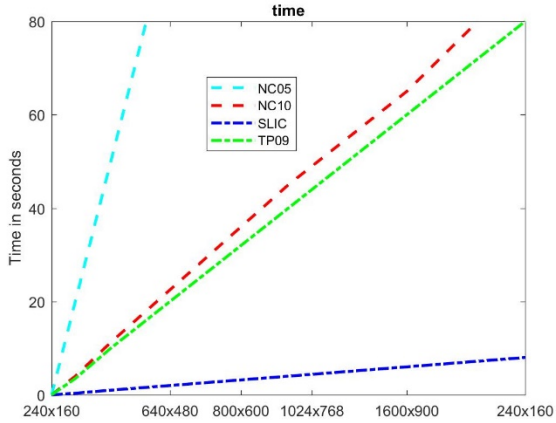

Figure 10. Time Complexity Comparison

3.2.2.2 Boundary recall [7]:

The performance of segmentation depends on the degree of adhering. If the segmentation boundaries are far away from graph true boundaries, the following fitting and rending processes will be hard to recover the original shape of the pictures. In this paper, two methods are used to evaluate the performance of segmentation: Boundary Recall and Under segmentation error.

Boundary Recall: Boundary recall measures the percentage of the ground truth edges fall within at least two pixels of a super-pixel boundary. As shown in the Figure 11, a base line is defined by uniform squares, labeled as 'Squares'. Among four segmentation algorithms, SLIC has the best performance. Followed by NC10 and NC 05, whose efficiency raise slowly with the increasing number of pixels. The TP09 has the worse segmentation performance.

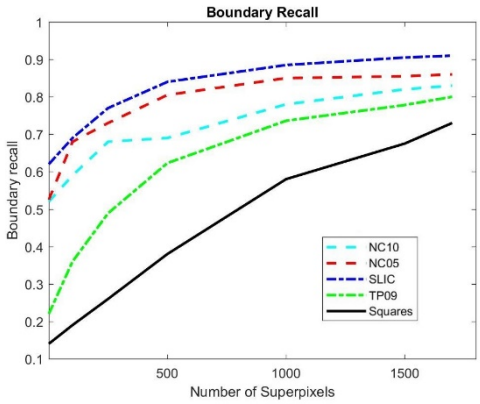

Figure 11. Boundary Recall

3.2.2.3 Under-segmentation Error [10]:

Another measure of boundary adherence is Under-segmentation error. The ground truth segmentation $g_i$and the set of super-pixels required to cover it, $S_j|S_j \cap g_i$,which can calculate the number of pixels across g_i. in the formula, B is the minimum pixels numbers in $S_j$and cross $g_i$and M is the number of pixels on the ground truth segmentations. In this project, B is set to 5% of numbers of segmentation pixels.

$$Error = \frac{1}{N}\left[\sum_{i=1}^{M}\left(\sum_{S_j|S_j \cap g_i}|S_j|\right) - N\right] \quad \textbf{(6)}$$

Shown in the Figure 12, uniform squares are set as the baseline. TP09 and NC05 have similar performance especially when the number of pixels becomes large. When pixels numbers over 300, SLIC has the best performance, however, when numbers of pixels are less than 300, NC10 has better segmentation effect.

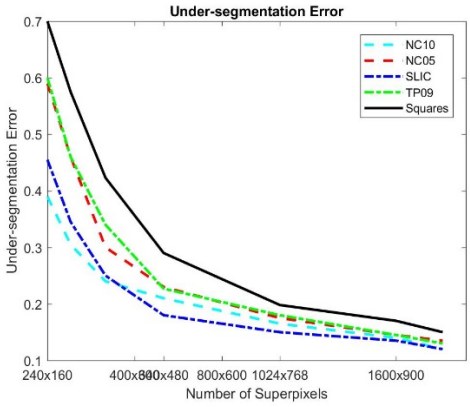

Figure12. Under-segmentation Error

4. Comparison of two segmentation methods

Among all the state-of-art segmentation methods we have tested before, SLIC segmentation method performs the best in all of our test attribute dimensions. Under this circumstances, we apply SLIC method to replace the original ncut multiscale segmentation method in Arty Shapes and list the comparison results as follows. The left part of following all figures is the result of using ncut multiscale segmentation method, while the right is result of applying SLIC method.

Figure 13. Original Image Input

(1) Segmentation: The boundary of SLIC segmented result is more smooth. But given the same hyper parameter N to both segmentation method, number of segments in SLIC result is smaller than N while N-cut is exactly equal to N. It enables N-cut to preserve more details of original image. Both segmentation methods achieve semantic results.

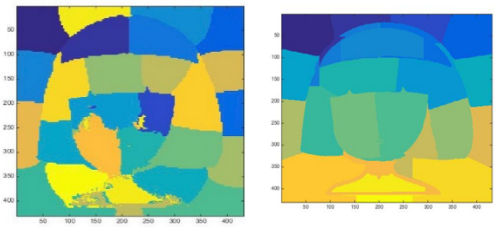

Figure 14. Segmentation comparison result

Left: MN-cut Right: SLIC

(2) Fitting Shapes on the segments

We fit triangles and robust convex hull to both segmented images in last step. Left is result of Ncut segmentation and Right part is SLIC result.

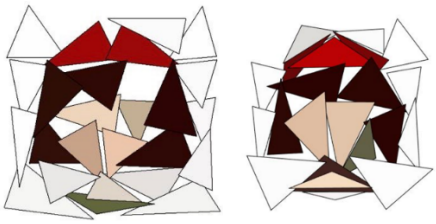

Figure 15. Fitting Triangles Comparison

Left: MN-cut Right: SLIC

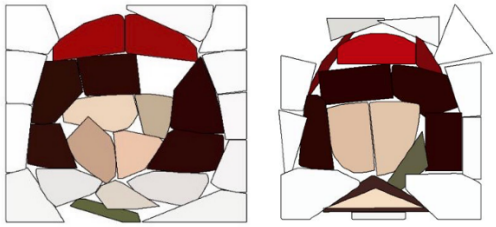

Figure16. Fitting Robust Convex Hull Comparison

Left: MN-cut Right: SLIC

(3) Automatic Shape Selection Comparison

From this automatic shape selection step we can see SLIC result produce a better abstraction than ncut segmentation method in representing the girl's outlook. SLIC rendering covers more semantically than MN-cut result.

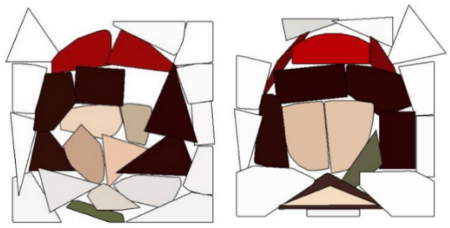

Figure17. Automatic Shape Selection Comparison

Left: MN-cut Right: SLIC

(4) Rendering

Figure 18 shows the result of setting threshold to 3 of ΔE to both of the images achieved from last step. While Figure 19 set threshold to 10. Using the same trained decision tree, SLIC rendering work acts better.

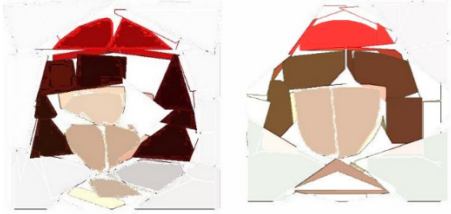

Figure 18. Rendering Comparison, ΔE = 3

Left: MN-cut Right: SLIC

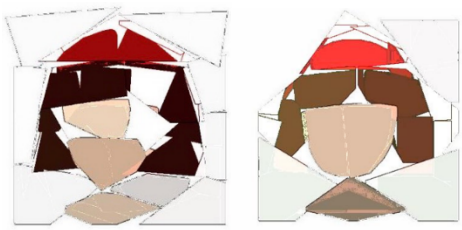

Figure 19. Rendering Comparison, ΔE = 10

Left: MN-cut Right: SLIC

5. Limitations & Future Work

Firstly, the second step of implementing arty shape abstraction technique is fitting shapes. Among the five shapes that we have chosen in before works. The time for creating robust convex hull is quite long, so the method of generating robust convex hull should be optimized in the future work.

What's more, the third step of implementing arty shape abstraction technique is automatic shape selection. We apply C4.5 decision tree in this step. However, as a greedy algorithm, once an attribute has been chosen as the node for a particular level of the tree, it does not reconsider this choice. It may lead to a locally optimal solution.

The last but not the least, we apply SLIC segmentation technique to improve arty shape abstraction technique. However, there's still some shortcomings of SLIC technique. Two hyper parameters are needed as input for SLIC segmentation, one is the number of clusters but it doesn't strictly equal to the number of parts in output rendering. The second parameter compactness factor needs adjustment for better performance which increases testing difficulty.

6. Conclusions

After implementing the Arty Shapes abstraction technique, we find the speed of segmentation process is quite slow which is inconvenient for practical use. We then test several state-of-art segmentation techniques and compare their performance in terms of under-segmentation error, boundary recall and time. SLIC segmentation method performs well among these four segmentation techniques of regarding to these three test dimension. We apply SLIC technique to replace ncut-multiscale segmentation technique in original work. The time of segmentation process is largely decrease and the final rendering work has better integrity in fitting the arty shapes from our experiment. The problem of SLIC technique is that the hyper parameter N that set as input doesn't strictly equal the number of blocks in output rendering.

Reference

[1] Song, Y. Z., Hall, P., Rosin, P., & Collomosse, J. (2008). Arty shapes. Computational Aesthetics 2008: 65-72

[2] Cour, T., Benezit, F., & Shi, J. (2005, June). Spectral segmentation with multiscale graph decomposition. In Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on (Vol. 2, pp. 1124-1131). IEEE.

[3] Voss, K., & Suesse, H. (1997). Invariant fitting of planar objects by primitives. IEEE Transactions on Pattern Analysis and Machine Intelligence, 19(1), 80-84.

[4] Rosin, P. L., & West, G. A. (1995). Curve segmentation and representation by superellipses. IEE Proceedings-Vision, Image and Signal Processing, 142(5), 280-288.

[5] Kolesnikov, A., & Franti, P. (2005, September). Optimal algorithm for convexity measure calculation. In Image Processing, 2005. ICIP 2005. IEEE International Conference on (Vol. 1, pp. I-353). IEEE.

[6] Sharma, G., & Bala, R. (Eds.). (2002). *Digital color imaging handbook*. CRC press

[7] Levinshtein, A., Stere, A., Kutulakos, K. N., Fleet, D. J., Dickinson, S. J., & Siddiqi, K. (2009). Turbopixels: fast superpixels using geometric flows. IEEE Transactions on Pattern Analysis & Machine Intelligence, 31(12), 2290-2297.

[8] Achanta, R., Shaji, A., Smith, K., Lucchi, A., Fua, P., & Süsstrunk, S. (2010). Slic superpixels. Epfl.

[9] Martin, D., Fowlkes, C., Tal, D., & Malik, J. (2002). A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. Computer Vision, 2001. ICCV 2001. Proceedings. Eighth IEEE International Conference on(Vol.2, pp.416-423 vol.2). IEEE.

[10] Veksler, O., Boykov, Y., & Mehrani, P. (2010). Superpixels and Supervoxels in an Energy Optimization Framework. Computer Vision - ECCV 2010 -, European Conference on Computer Vision, Heraklion, Crete, Greece, September 5-11, 2010, Proceedings (Vol.6315, pp.211-224). DBLP.