Breaking Art: Synthesizing Abstract Expressionism Through Image Rearrangement

Christopher Palazzolo[a], Oliver van Kaick[a], David Mould[a,*]

[a]*Carleton University, 1125 Colonel By Dr, Ottawa, K1S 5B6, Ontario, Canada*

Abstract

We present an algorithm that creates interesting abstract expressionist images from segments of an input image. The algorithm operates by first segmenting the input image at multiple scales, then redistributing the resulting segments across the image plane to obtain an aesthetic abstract output. Larger segments are placed using neighborhood-aware descriptors, and smaller segments are arranged in a Poisson disk distribution. In our thorough analysis, we show that our results score highly according to several relevant aesthetic metrics, and that our style is indeed abstract expressionism. The results are visually appealing, provided the exemplar has a somewhat diverse color pallette and some amount of structure.

Keywords: Abstract Art Synthesis, Image Generation, Image Segmentation

1. Introduction

Automatically generating abstract artwork has a number of applications, such as quickly creating hundreds of unique images for populating homes and museums in virtual worlds. Of course, the automatically created images may be appealing on their own merits. Previous work has proposed methods to transfer artistic styles to photographs [1], generate "tidied up art" [2, 3], or provide the tools to manually create artwork [4, 5]. However, no previous approach enables automatic generation of abstract expressionism. In this paper, our objective is to create abstract images similar to abstract expressionism, which we accomplish by rearranging elements from a user-provided input image.

Our approach synthesizes an image in three layers: coarse segments, fine segments, and a background gradient field. We use flood fill to extract segments from an input photograph, inheriting its color distribution. In the synthesis phase, the coarse segments are assembled using a neighborhood-descriptor-based system, and the fine segments and gradient field layers are created with a Poisson disk distribution [6]. Figure 1 showcases a sample of the rich generative space of our algorithm on display in a virtual art gallery.

Our method produces an original style of abstract expressionism art with aesthetic attributes comparable to those of historical artwork. Our goal is to create aesthetic objects with little overhead. Because we rearrange patches of content from the exemplar, one might imagine that our work is similar to patch-based texture synthesis algorithms such as image quilting [7]. However, our work differs both in process and intent. Patch-based synthesis methods aim to synthesize local patterns that resemble an exemplar, whereas we use non-texture exemplars and make no attempt to preserve their large-scale structure in our segment placement process. The exemplars mainly provide the color scheme of the result and we do not expect to synthesize visually similar images.

Our contributions are as follows:

- We present a novel algorithm to synthesize attractive abstract images.
- We demonstrate that we generate images in the desired style (abstract expressionism), and that the resulting images are similar to existing works of the same style, according to aesthetic measures.
- We find the most visually similar existing artworks to our results, and show that they are quite distinct

*Corresponding Author
Email addresses:
ChristopherPalazzolo@cmail.carleton.ca (Christopher Palazzolo), OlivervanKaick@cunet.carleton.ca (Oliver van Kaick), DavidMould@cunet.carleton.ca (David Mould)

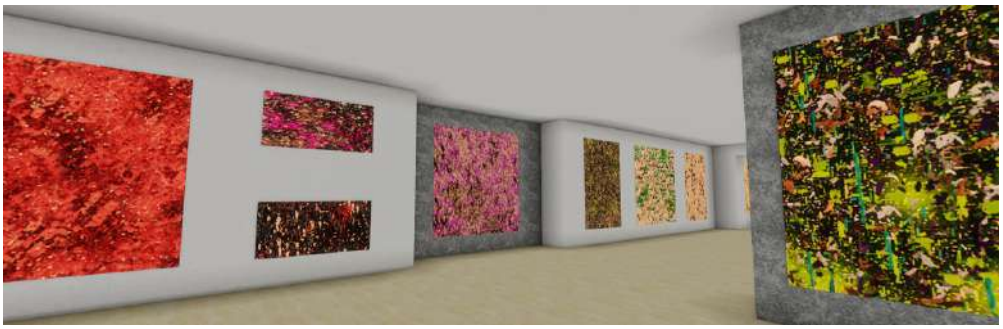


Figure 1: A virtual 3D art gallery showcasing some of our results.

both visually and in terms of the selected perceptual similarity metrics.

Our source code can be downloaded at https://github.com/ChrisPGraphics/BreakingArt

2. Related Work

We discuss two areas of research: algorithmic synthesis of art, particularly abstract art; and analysis of images with a focus on quality assessment and categorization, which we use for our analysis.

2.1. Artistic Image Synthesis

Using neural networks to synthesize stylized images has received considerable attention. Tan et al. [8] build on previous conditional and categorical GAN-based approaches to synthesize artwork through backpropagating the image labels to the generator in the loss function. Yi et al. [9] show that results of painting synthesis can be improved by using a diffusion model. Pérez and Cozman [10] train a GAN to synthesize paintings in different art styles, potentially usable for data augmentation in future art style classification work. While the results are impressive, these methods are not intended for abstract art and their results here are less satisfactory.

A number of authors including Gatys et al. [1] explore style transfer, the task of rendering the content of one image in the style of a second. One might conclude that this could be used to synthesize abstract art by using a real abstract painting as the style image. We emphasize that this is not the same objective: style transfer preserves global structure by changing low-level details, whereas we preserve the low-level details to synthesize a novel global structure to create a new aesthetic object.

Lee et al. [4] and Alvarez et al. [5] propose methods that allow the user to manually create abstract artwork. This is in contrast to our algorithm which is fully automatic. Lee et al. simulate real-time 3d fluid jets of paint whereas Alvarez et al. allow the user to combine geometric shapes in a tree data structure.

Zhao et al. [11] generate abstract art from photographs by using semantic segmentation and introducing ambiguity to each element with a range of operators. Yan et al. [12] use brush strokes to render the photorealistic exemplar as a painting. We perform the opposite process: our method uses low-level details from the exemplar and generates a novel global structure.

The closest work to ours is the work of Ufer et al. [2] and Gieseke et al. [3], who independently proposed algorithms inspired by the book by comedian Ursus Wehrli [13]. Wehrli proposed "tidying up art": taking the distinct elements of an abstract painting, and rearranging them in a more organized manner. The algorithmic methods proceed similarly. Starting from image segmentation, they then use a custom element arrangement procedure that can be tuned to get the desired result. Our general process is similar, but we aim to create a more stochastic result instead of a tidy one. Section 4 further discusses the difference.

2.2. Art Analysis

There is a substantial body of literature on automated assessment of images, primarily for photographic images but sometimes also applied to art. Sartori et al. [14] introduce the MART database, and use it for training a method to predict and analyze the emotions experienced by a person observing artwork. Malu et al. [15, 16] train a multi-task CNN to predict the aesthetic score of a photograph, along with a number of other metrics to justify that decision. Their model reliably predicts aesthetic attributes compared with scores in the Aesthetics Attribute Database [16]. We use the model by Malu et al. to compare with historical abstract paintings.

Lecoutre et al. [17] train a model using the WikiArt database [18] to classify the style of art that a painting belongs to, achieving an overall accuracy of 62%,

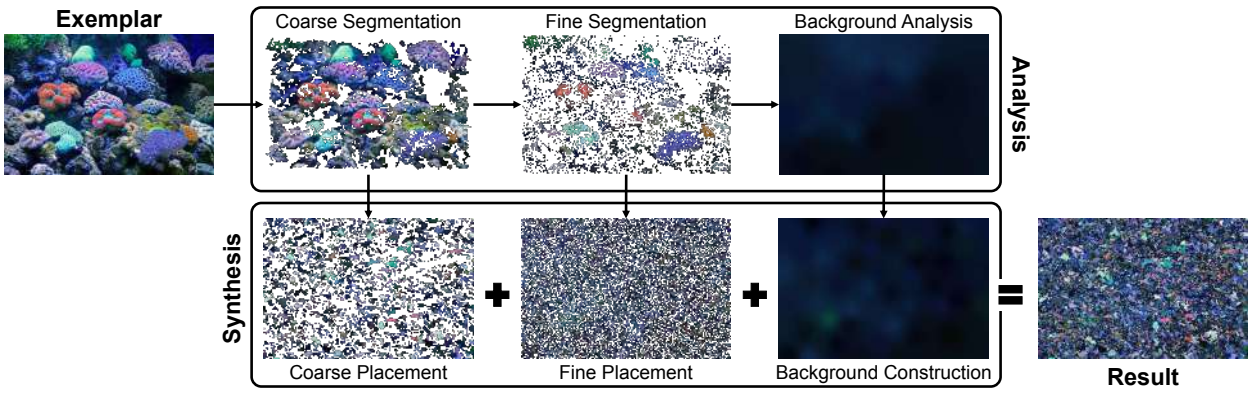


Figure 2: A high-level overview of our synthesis process. There are two phases: analysis of an input exemplar, then synthesis of a new image.

surpassing all previous state-of-the-art models. We use this model to categorize the genre of images produced by our algorithm.

3. Breaking Art

Abstract expressionism is a broad category of artwork. Our goal is to emulate a specific subcategory of paintings that are made of a composition of irregular and largely unrecognizable elements with a high amount of color coherence. We accomplish this by breaking up an input image into nonuniform polygons using flood fill. This ensures each polygon contains a fairly regular color. Synthesizing an image based on these polygons causes the result to look abstract and maintains a similar color palette to the exemplar.

Other sub-styles, such as color field paintings, are not well emulated by our algorithm. While both have regions of a solid color, our process uses small irregular fragments of color whereas color field paintings have far larger and more regular regions; often a region is a significant fraction of the entire image, 30% or more.

We designed our algorithm to be efficient and simple to use. For efficiency, our algorithm extracts local structure in the form of segments from an exemplar and spreads it across a canvas without preserving the global structure of the exemplar. The segments are placed so that they loosely preserve some properties of the exemplar, which makes the result more appealing than placing elements at random; the viewer can appreciate that some structure is present. Our method is easy to use and fully automatic as the only input is an exemplar that controls the color palette and local structure of the output. We do not require any tedious tuning of parameters.

Figure 2 shows the high-level structure of our algorithm. We use an analysis phase, dividing the exemplar into segments, and a synthesis phase, where the segments are used to construct an output texture. The segments extracted from the exemplar have little semantic meaning. The algorithm attempts to learn a pattern from the distribution of the extracted polygons, even though none exists. The analysis step instead hallucinates a pattern in the distribution which it then attempts to reproduce as the result. This pattern hallucination is what creates the color and shape distributions in the output.

Our analysis step decomposes the exemplar into elements to be rearranged during synthesis. We have three types of segments, plus background, as shown in Figure 3. The *coarse segments* are the larger and more prominent features of the exemplar. *Fine segments* are the smaller and less noticeable features between the coarse segments. We also have *detail polygons*, each of which is associated with a particular coarse segment. Pixels that are not part of any coarse or fine segment are used to define a background gradient field to complete the image. All layers are crucial for creating a complete, visually interesting result.

In the subsequent synthesis step, we use an iterative polygon placement algorithm based on neighborhood descriptors to create the coarse segment layer, and a simple Poisson disk distribution for the fine segments and background gradient field. These three layers are then merged together to produce the result. More details of both phases are given in the following subsections.

3.1. Analysis Phase

Coarse segment extraction. In the analysis step, we start with an exemplar with color channels normalized between 0 and 1. First, we extract the coarse segments using flood fill (tolerance 0.2). Segments smaller than 10 pixels or larger than 300 (thresholds chosen empiri-

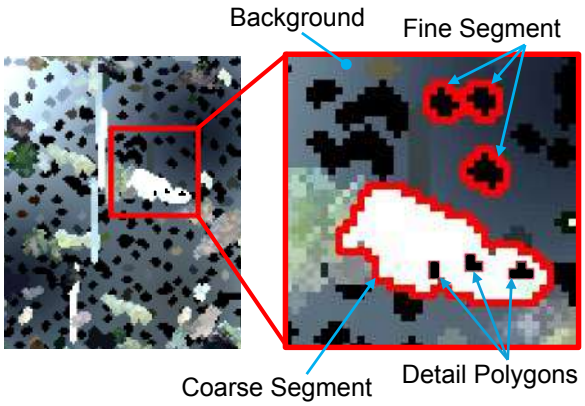


Figure 3: Some of the three types of segments and the background gradient.

cally) are discarded, ensuring only meaningful and significant segments are kept. Flood fill segmentation proceeds incrementally, with the seed point for the next region being the first non-labeled pixel in a raster order traversal of the image.

While a vast array of segmentation techniques are available, flood fill is well suited to our purposes, as it produces non-semantic segments with irregular structures, advantageous in producing more chaotic abstract images. Flood fill often breaks up objects that have diverse colors into diverse and non-uniform shapes, whereas more semantically-aware methods would preserve the objects' structure. Figure 5 shows a comparison of alternate segmentation strategies. Comparisons using additional exemplars can be found in the supplemental material.

Following the initial segmentation, we compute the median color of all pixels contained within each coarse segment and use it as the segment's background color. Next, we apply a second pass of flood fill inside each coarse segment with a smaller tolerance (0.05 in the results shown) to obtain the detail polygons. The detail polygons are considered children of the coarse segment they are found in and provide extra visual detail. A detail polygon's color is the median color of its pixels.

Clustering and descriptor construction. Next, we organize the coarse segments into clusters based on visual similarity and build a descriptor for each segment. We use the color, area, and Polsby-Popper compactness [19] as features, and apply K-Means to get the clusters. We found that 15 clusters produce good results. A descriptor is a square-shaped region around a segment that characterizes the distribution of other segments in the vicinity of the central segment. It is stored as a 2D array at pixel resolution where each cell contains the cluster ID of the polygon type that appeared there in the exemplar. We will use the descriptors to compute scores based on spatial overlaps when synthesizing a new distribution, approximately preserving the local arrangement of the coarse segments in the result.

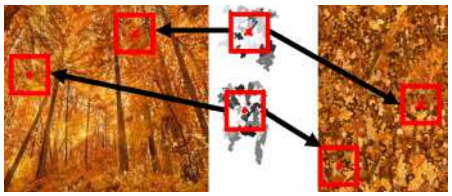


Figure 4: An example of segments and their descriptors. Left: The segments found in the exemplar in their original context. Center: the extracted segments and their descriptors (the red squares). Right: The location of the segments in the synthesized result.

When creating descriptors, we want to ensure each descriptor has enough segments in any direction to preserve low-level structure, but not too many to risk making a copy. We found that an average of 2.25 segments in any direction produces good results. This way of defining descriptor sizes also ensures our method is scale-independent. Segments touching the descriptor square are included in the descriptor. Figure 4 shows examples of descriptors of two segments and their locations in the exemplar and result.

Fine segment extraction. To obtain the fine segments, we apply flood fill on all pixels not previously marked as part of a coarse segment (tolerance 0.1 and must have an area between 3 and 300). We compute the median color of the contained pixels to get the segment's color. We do not attempt to find detail polygons for these segments, nor do we compute descriptors, as fine segments are intended to capture unstructured content of the exemplar.

Background extraction. Finally, we extract a color palette from the remaining pixels, i.e., those not part of any coarse or fine segment. The palette will be used to create a background gradient field at synthesis time. Our process begins by placing points according to a Poisson disk distribution over the entire image (spacing 50 pixels, chosen empirically); we then construct the Voronoi diagram of this point set. For each Voronoi region, we take the median color of all pixels that are not part of a coarse or fine segment and add that as a palette entry.

The analysis phase is now complete. For the coarse segments, we have the neighborhood of each segment as its descriptor, a cluster that the segment belongs to, and detail polygons to add more visual interest to the segment. We also have the fine segments and their median inter-element spacing. Finally, we have a color palette

extracted from the background which can be used to construct the background gradient field.

3.2. Synthesis Phase

We synthesize the result in three stages: a layer of coarse segments; a layer of fine segments; and a background gradient field. These three layers are then merged to produce the final result.

Coarse segment layer. The pseudocode for this process can be found in the supplemental material. The coarse segments are placed incrementally by choosing segments at random and scoring the placement of each one. Scores are based on the degree to which an attempted placement aligns with the descriptors of previously-placed segments.

We maintain a grid map, the same size and resolution as the target output, to store the evolving arrangement of finalized and potential segments. Grid cells can store values of one of four types: "undefined", meaning that nothing has been placed in the cell yet; "empty", meaning that the cell is currently empty and should remain empty; "finalized", with an accompanying category (cluster) ID for the segment that covers that grid cell; or "requested", with an accompanying category ID for the segment type that ought to be placed there.

A randomly placed segment S is scored based on its overlap with cells in different categories; if the grid cell requested a segment with S's category, the score increases, while mismatches or overlaps with non-requesting cells cause the score to decrease.

An attempt at segment placement involves first choosing a random grid cell with "requested" value. The probability distribution comes from a distance transformation on the requested cells to give more weight to centers of large clusters. Next, we pick a random segment of the requested category, place it, and computing its score. The algorithm then attempts to locally optimize the position, shifting the segment centre one pixel at a time as long as doing so improves the score. We make k attempts (10 in the results shown) and finalize the placement of the top-scoring segment, updating the grid map: all "undefined" entries nearby are given "empty" or "requested" values according to the descriptor of the placed segment, and the grid cells beneath the segment itself are given the value "finalized".

After synthesis of the layer is complete, we use global constraints to clean up the result. First, we measure how much of the exemplar is covered by each category of polygon. For each polygon category, we remove polygons at random until the category coverage of the result drops below the expected coverage from the exemplar. To add some randomness to the result, the polygon responsible for causing the coverage to drop below the threshold is only removed 50% of the time to ensure we are not always under the expected coverage.

Fine segment layer. We take the median of the inter-element distances of the fine segments from the exemplar and construct a Poisson disk distribution [6]. At each of these points, if it is not already covered by a coarse segment, we place a randomly-selected fine segment.

Background layer. We generate a set of control points in a Poisson disk distribution, and assign to each a random color from the exemplar palette. Using the combination of control points and colors, we interpolate the entire field using Inverse Distance Weights interpolation [20].

Layer merging. The three layers are then merged by drawing the background gradient field followed by the fine and coarse segments. Drawing the coarse segments last ensures they are not occluded by the other layers. All segments are opaque: the last-drawn segment overwrites any pixel it covers.

4. Results

Figure 6 gives results from our method, showing both the input exemplar and the synthesized abstract image. Figure 7 shows some additional results where the names are puns. Drastically different results can be obtained depending on the structure and color distribution of the exemplar.

Figure 5 shows some alternative segmentation and placement strategies. Alternative segmentation strategies include SLIC [21], uniform circular patches, and irregular random-shaped patches. Alternative segment placement strategies include using random placement, and placement according to a Poisson Disk distribution [6]. This figure also shows the importance of the fine segments and gradient field which are not included in the bottom two rows. SLIC and uniform patch extraction produce segments with little size and shape variation and the result is mundane. Irregular patches contain too many colors, with incoherent results.

Figure 8 highlights the difference between the objectives of our work, and those of "tidying up art". Both Ufer et al. [2] and Gieseke et al. [3] break up an image to rearrange the components in a more organized manner, whereas we rearrange the components in a more stochastic arrangement to create an aesthetic object.

We next turn our attention to comparisons between the results of our algorithm and artwork created by historical artists. We use the model of Lecoutre et al. [17]

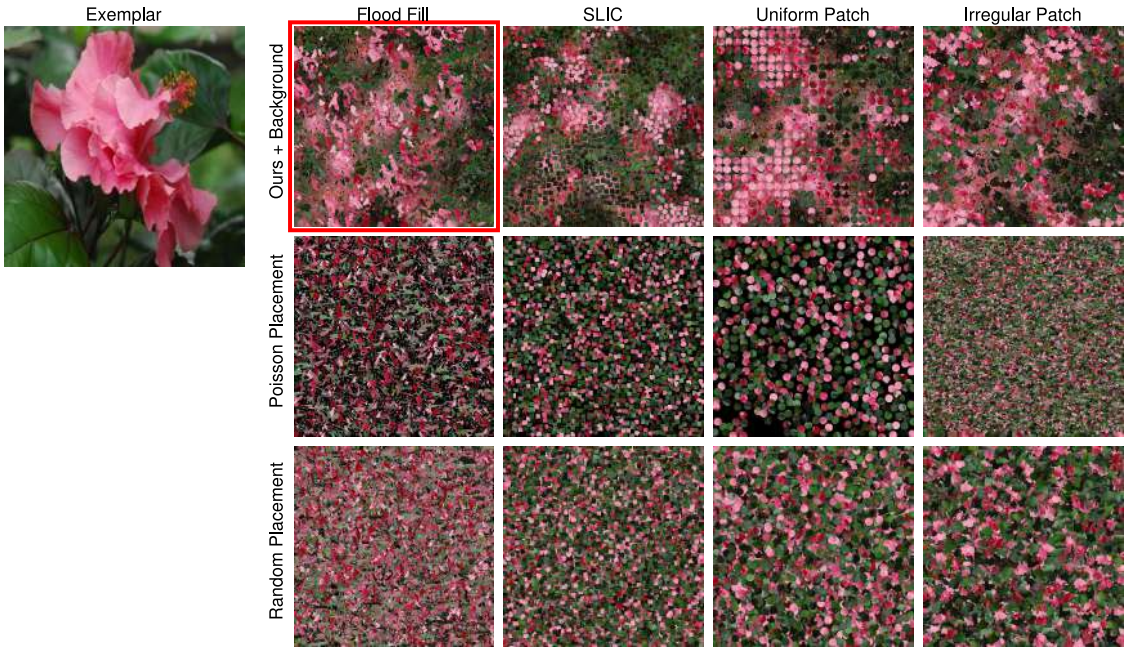


Figure 5: Alternative segmentation and placement strategies for synthesizing images, shown across different columns and rows, respectively. Our result (top left) is highlighted in red.

to confirm the art style produced by our algorithm. We then use the model of Malu et al. [15] to compare the aesthetic metrics of our results with images produced by historical artists. Finally, we use a number of perceptual similarity metrics to compare our work with existing abstract expressionism to show that our images are visually distinct compared to existing artwork in the genre.

The Aesthetics Attribute Database [16] is a database of nearly ten thousand photographs containing a wide range of subjects, structure, and colors. From this, we took 500 random images, and synthesized results using them as exemplars. We compare these results with paintings from the WikiArt database [18] in the following subsections.

4.1. Art Style Classification

We use the pretrained model provided by Lecoutre et al. [17] to categorize 500 images produced by our method. More than 50% of these images were classified as abstract expressionism, far more than any other category. Other categories in the top ten include abstract art and art informel which can be thought of as European abstract expressionism [17]. We can reasonably conclude that our algorithm does indeed produce abstract expressionist images. The breakdown of the top ten categories can be found in the supplemental material.

4.2. Aesthetic Metrics

We compared our set of 500 images with nearly 3000 abstract expressionism artworks from the WikiArt database [18] using the model trained by Malu et al. [15], and recorded the scores for the five attributes and final aesthetic score for each image. It should be noted that this model was trained on photographs, not artwork. To our knowledge this is the most reliable aesthetic image evaluator.

Malu et al. estimate five aesthetic attributes: *color harmony*; *interestingness of content*; *depth of field*; *object emphasis*; and *vivid color*. The estimated values are in the range {-1,1}; a value of -1 means that the attribute reduces the photograph's appeal, and +1 indicates the attribute improves the visual appeal. They also compute a final *aesthetic score*, ranging between 0 and 1. The distribution of scores for each metric, for our images and a sampling of images from the database, are shown in Figure 9.

According to the model, the color harmony of our images contributes significantly to their overall visual appeal. Conversely, the low object emphasis in our images reduces their appeal. This makes sense because our algorithm's art style does not have discernible objects in the result, whereas many historical paintings include recognizable objects, even if represented in an abstract way. The remainder of the histograms are very simi-

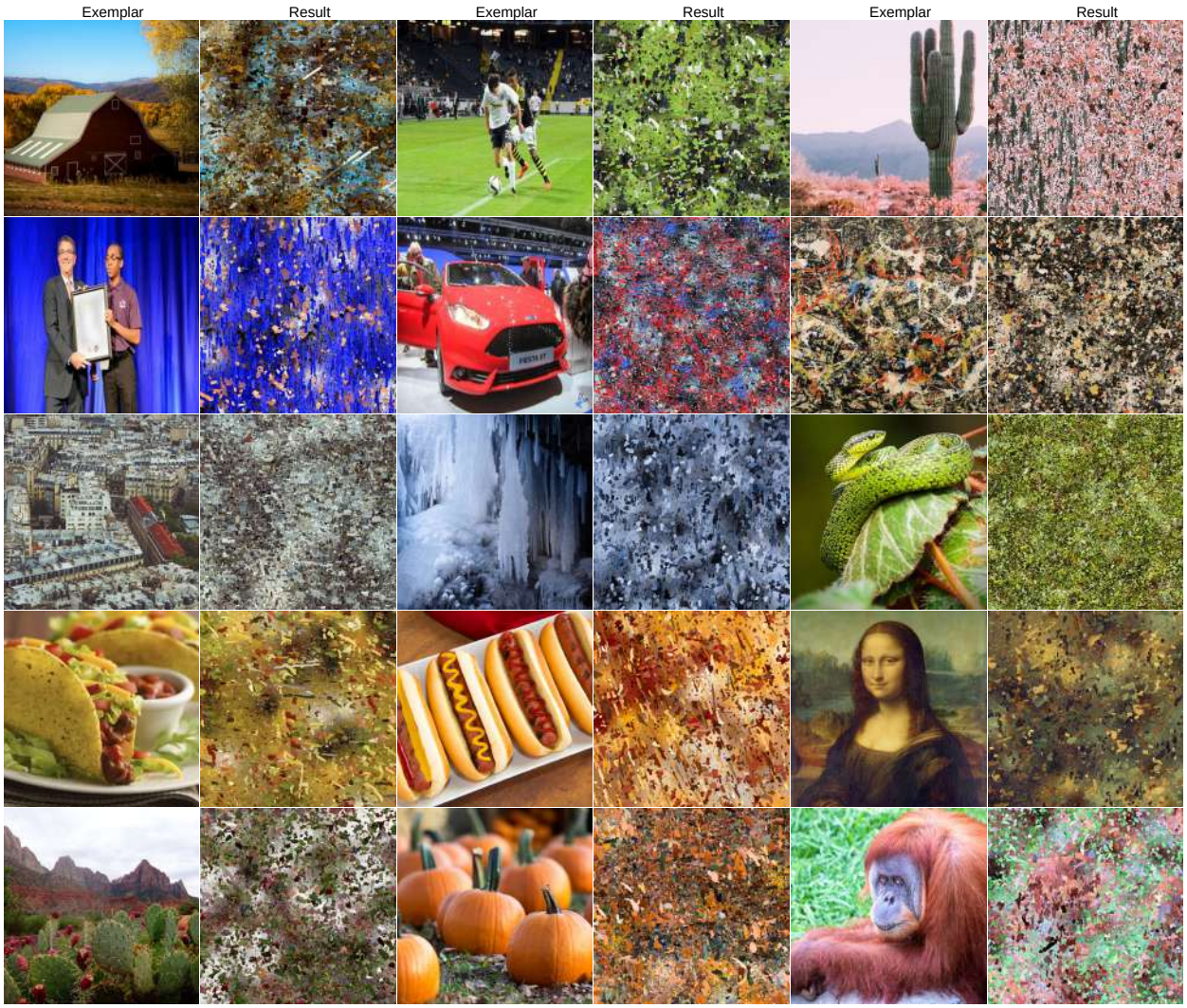


Figure 6: A sample of images synthesized by our method from the given exemplars.

Figure 7: A sample of images synthesized by our method. The name given to each image is a pun related to the content of the exemplar.

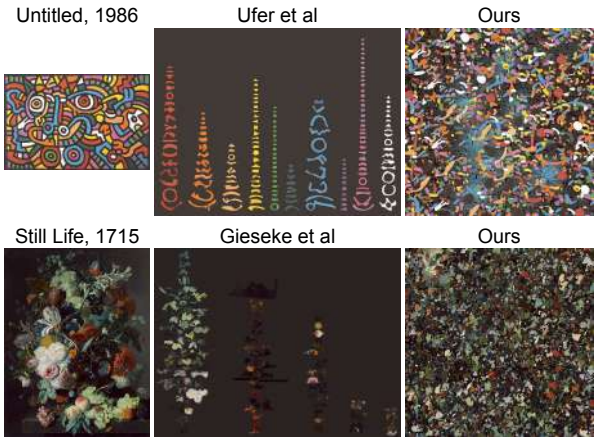


Figure 8: A comparison between our work, and the "tidying up art" papers of Ufer et al. [2] and Gieseke et al. [3] to demonstrate the opposite objectives or our work and theirs.

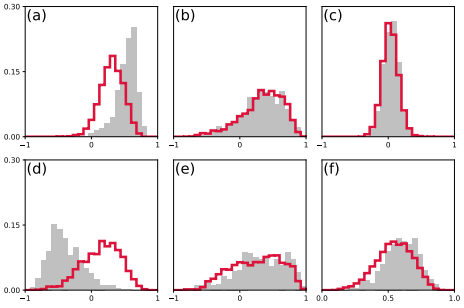
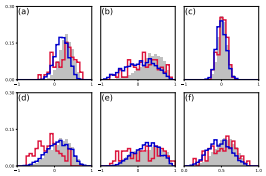


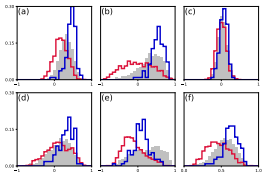
Figure 9: The distribution of our images (solid gray) compared to abstract expressionism paintings from the WikiArt database [18] (red outline) for five criteria, plus an aggregate score: (a) Color Harmony, (b) Interestingness of Content, (c) Depth of Field, (d) Object Emphasis, (e) Vivid Color, and (f) Aesthetic Score.

lar. Overall, although our artwork is part of the abstract expressionism distribution, it does not span the full distribution, being generally more abstract than historical images in this genre. The highest and lowest scoring image for each aesthetic attribute can be found in the supplemental material.

Figure 10a compares abstract expressionism against two other recognized subsets, color field and action paintings. Comparing these distributions to those of Figure 9, one can see that our results are roughly as close to abstract expressionism as the other subcategories. As a control, Figure 10b compares the distribution of abstract expressionism and two different styles, minimalism and analytical cubism. With the exception of depth of field, each art style has a distinct histogram pattern.



(a) Closely related styles: Action Painting (red), Color Field Painting (blue).



(b) Distinct styles: Minimalism (red), Analytical Cubism (blue).

Figure 10: A comparison between abstract expressionism paintings from the WikiArt database [18] (solid gray) and other art styles using the same criteria as Figure 9.

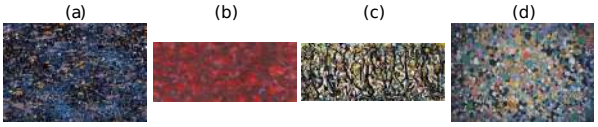


Figure 11: The most similar artworks to the results of our method based on four perceptual similarity metrics. (a) Our result and the query image, (b) best DISTS and SSIM scores (0.252 and 0.630, respectively), (c) best LPIPS score (0.352), (d) best PieAPP score (0.408). With DISTS, LPIPS and PieAPP, 0 is a perfect match, and SSIM has 1 as a perfect match and 0 as no similarity.

4.3. Visual Similarity

The previous subsections show that our algorithm is capable of producing images resembling abstract expressionism. Here, we consider how similar our results are to the styles of historical artists. To do this, we select a random image produced by our algorithm, and find the image in the abstract expressionism category in the WikiArt database that has the best perceptual similarity score. We use the metrics DISTS [22], LPIPS [23], PieAPP [24], and SSIM for this comparison, using the implementation of these metrics provided by the PyIQA Python library [25]. Figure 11 shows the results for one image; additional results are shown in the supplemental material.

Although we used perceptual similarity metrics to find similar paintings to our generated images, none possess a similar style or element distribution to ours. The result in Figure 11 is typical; in no cases were the historical artworks visually similar to our results. This provides some evidence that our algorithm produces visually distinct images compared to historical abstract expressionism.

4.4. Successful Exemplar Selection

From analyzing hundreds of exemplars synthesized by our algorithm, we observed that two attributes have the biggest impact on the quality of results: color and structure.

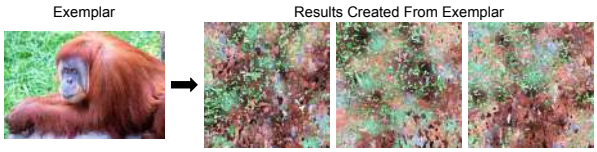


Figure 12: A few images synthesized from the same exemplar to demonstrate the low diversity in the results.

Color contrast is one of the most important attributes informing images aesthetics. While the results in the bottom row of Figure 13 look interesting individually, they are less attractive than the images in the top row. The exemplars of the bottom row are all grayscale images. The difference between the highest and lowest scoring result in Figure 13 seems to mostly manifest in colors, where red, orange and pink make a more exciting color palette than gray, white and black.

Moreover, the exemplar must have some structure. We tried a number of smooth gradients as exemplars; these failed to produce good results because the coarse and fine segmentation steps yielded few if any distinct segments. In this case, the result is completely created from the background gradient field, with unappealing results.

4.5. Performance

Analysis takes an average of 2 minutes for a 500×500 exemplar. Synthesis takes on average 45 seconds to produce a 500×500 result and 6.5 minutes for a 1000×1000 result. These values can vary depending on the number of identified segments and the size of the exemplar.

4.6. Limitations

We observed low diversity between results synthesized from the same exemplar. To generate a diverse set of artwork, the user requires an equally large set of distinct exemplars. However, there are massive databases available of diverse photographs and images, mitigating this concern. Figure 12 shows an example of this. The results are not exactly alike, but they share a similar structure and color palette.

Grayscale exemplars do not score very highly according to the model trained by Malu et al. [15]. The fifty lowest scoring images are all based on grayscale exemplars. We somewhat disagree with the model's conclusion, since these images are aesthetically pleasing individually; however, when viewing multiple results of grayscale exemplars together, they appear boring and repetitive. This indicates that diverse color is an important factor in constructing a collection of images.

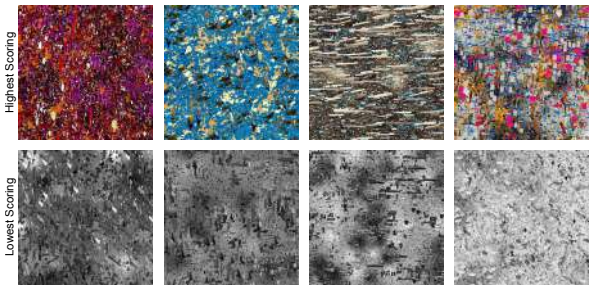


Figure 13: The four highest-scoring (above) and lowest-scoring results (below), according to the model trained by Malu et al. [15].

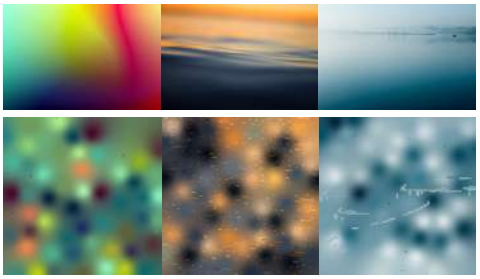


Figure 14: Images synthesized from exemplars lacking distinct features. Above: exemplar; below: our result.

Exemplars without distinct features, such as gradient images, do not produce good results because either no segments are extracted, or the segments are too large and cannot be used for synthesis. Figure 14 shows three failure cases from exemplars with little structure.

Finally, our method produces a specific style of output. It is not a general purpose algorithm like style transfer techniques. Instead, it focuses on producing images in a specific and novel style in a small amount of time.

5. Conclusion

We introduced an image synthesis algorithm which is capable of producing abstract expressionism images with similar aesthetic scores as artworks created by historical artists. We construct an abstract image from an input photograph, with an analysis step followed by a synthesis step. The analysis deconstructs the exemplar into coarse and fine segments, and identifies colors that can be used to form a background gradient mesh. The synthesis step creates a coarse segment layer from the coarse segments, a detail layer from the fine segments, and a background layer using the extracted background colors; the three layers are merged to produce the abstract output.

The choice of exemplar is important to ensure the result will be comprised of an exciting color palette and

contain enough structure to be visually appealing.

Future work could include segment augmentation and other diversification strategies so that a single exemplar can produce distinct results. With this change, less images would be required for generating a large library of artwork. Another avenue of future work could explore more expensive polygon placement strategies such as through the use of an optimizer. Our texton placement strategy avoids collisions, and ensures local structure is preserved with little overhead; however, an optimizer may provide better results at the cost of significantly more synthesis time.

Acknowledgments

We thank the anonymous reviewers for their valuable feedback. This work was partially supported by the Natural Sciences and Engineering Research Council of Canada (NSERC) and Carleton University. We would also like to thank the members of the Graphics, Imaging, and Games Lab (GIGL) as well as our friends outside the lab for their suggestions and comments.

References

[1] L. Gatys, A. Ecker, M. Bethge, A neural algorithm of artistic style, Journal of Vision 16 (12) (2016) 326. doi:10.1167/16.12.326.

[2] N. Ufer, M. Souiai, D. Cremers, Wehrli 2.0: An algorithm for "tidying up art", in: A. Fusiello, V. Murino, R. Cucchiara (Eds.), Computer Vision – ECCV 2012. Workshops and Demonstrations, Springer Berlin Heidelberg, Berlin, Heidelberg, 2012, pp. 532–541.

[3] L. Gieseke, S. Klingel, M. Fuchs, Shake it up - image decomposition and rearrangements of its constituents, in: Proceedings of the Workshop on Computational Aesthetics, CAE '15, Eurographics Association, Goslar, DEU, 2015, p. 141–147.

[4] S. Lee, S. C. Olsen, B. Gooch, Interactive 3d fluid jet painting, in: Proceedings of the 4th International Symposium on Non-Photorealistic Animation and Rendering, NPAR '06, Association for Computing Machinery, New York, NY, USA, 2006, p. 97–104. doi:10.1145/1124728.1124745.

[5] L. Alvarez, N. Monzón, J.-M. Morel, Interactive design of random aesthetic abstract textures by composition principles, Leonardo 54 (2) (2021) 179–184. doi:10.1162/leon_a_01768.

[6] M. McCool, E. Fiume, Hierarchical Poisson disk sampling distributions, in: Proceedings of Graphics Interface '92, GI '92, Canadian Human-Computer Communications Society, Toronto, Ontario, Canada, 1992, pp. 94–105.

[7] A. A. Efros, W. T. Freeman, Image Quilting for Texture Synthesis and Transfer, 1st Edition, Association for Computing Machinery, New York, NY, USA, 2001.

[8] W. R. Tan, C. S. Chan, H. E. Aguirre, K. Tanaka, ArtGAN: Artwork synthesis with conditional categorical GANs, in: 2017 IEEE International Conference on Image Processing (ICIP), 2017, pp. 3760–3764. doi:10.1109/ICIP.2017.8296985.

[9] D. Yi, C. Guo, T. Bai, Exploring painting synthesis with diffusion models, in: 2021 IEEE 1st International Conference on Digital Twins and Parallel Intelligence (DTPI), 2021, pp. 332–335. doi:10.1109/DTPI52967.2021.9540115.

[10] S. P. Pérez, F. G. Cozman, How to generate synthetic paintings to improve art style classification, in: A. Britto, K. Valdivia Delgado (Eds.), Intelligent Systems, Springer International Publishing, Cham, 2021, pp. 238–253.

[11] M. Zhao, S.-C. Zhu, Abstract painting with interactive control of perceptual entropy, ACM Trans. Appl. Percept. 10 (1) (Mar. 2013). doi:10.1145/2422105.2422110.

[12] M. Yan, Y. Pu, P. Zhao, D. Xu, H. Wu, Q. Yang, R. Wang, Abstract painting synthesis via decremental optimization, Computer Graphics Forum 41 (7) (2022) 419–430. doi:https://doi.org/10.1111/cgf.14688.

[13] U. Wehrli, A. G. von Olenhusen, Kunst aufräumen, Kein & Aber, 2002.

[14] A. Sartori, D. Culibrk, Y. Yan, R. Job, N. Sebe, Computational modeling of affective qualities of abstract paintings, IEEE MultiMedia 23 (3) (2016) 44–54. doi:10.1109/MMUL.2016.20.

[15] G. Malu, R. S. Bapi, B. Indurkhya, Learning photography aesthetics with deep CNNs, CoRR abs/1707.03981 (2017). arXiv:1707.03981.

[16] S. Kong, X. Shen, Z. Lin, R. Mech, C. Fowlkes, Photo aesthetics ranking network with attributes and content adaptation, in: European Conference on Computer Vision (ECCV), 2016.

[17] A. Lecoutre, B. Negrevergne, F. Yger, Recognizing art style automatically in painting with deep learning, in: M.-L. Zhang, Y.-K. Noh (Eds.), Proceedings of the Ninth Asian Conference on Machine Learning, Vol. 77 of Proceedings of Machine Learning Research, PMLR, Yonsei University, Seoul, Republic of Korea, 2017, pp. 327–342.

[18] W. R. Tan, C. S. Chan, H. Aguirre, K. Tanaka, Improved ArtGAN for conditional synthesis of natural image and artwork, IEEE Transactions on Image Processing 28 (1) (2019) 394–409. doi:10.1109/TIP.2018.2866698.

[19] D. Polsby, R. Popper, The third criterion: Compactness as a procedural safeguard against partisan gerrymandering (2015).

[20] D. Shepard, A two-dimensional interpolation function for irregularly-spaced data, in: Proceedings of the 1968 23rd ACM National Conference, ACM '68, Association for Computing Machinery, New York, NY, USA, 1968, p. 517–524. doi:10.1145/800186.810616.

[21] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, S. Süsstrunk, SLIC superpixels compared to state-of-the-art superpixel methods, IEEE Transactions on Pattern Analysis and Machine Intelligence 34 (11) (2012) 2274–2282. doi:10.1109/TPAMI.2012.120.

[22] K. Ding, K. Ma, S. Wang, E. P. Simoncelli, Image quality assessment: Unifying structure and texture similarity, CoRR abs/2004.07728 (2020). arXiv:2004.07728.

[23] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, O. Wang, The unreasonable effectiveness of deep features as a perceptual metric, in: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2018, pp. 586–595. doi:10.1109/CVPR.2018.00068.

[24] E. Prashnani, H. Cai, Y. Mostofi, P. Sen, PieAPP: Perceptual image-error assessment through pairwise preference (2018). arXiv:1806.02067.

[25] C. Chen, J. Mo, IQA-PyTorch: Pytorch toolbox for image quality assessment, [Online]. Available: `https://github.com/chaofengc/IQA-PyTorch` (2022).