Vision, Modeling, and Visualization (2023)
T. Grosch and M. Guthe (Eds.)

Semantic Image Abstraction using Panoptic Segmentation for Robotic Painting

Michael Stroh[1], Jörg-Marvin Gülzow[2], and Oliver Deussen[2]

[1,2,3]University of Konstanz, Germany

Abstract

We propose a comprehensive pipeline for generating adaptable image abstractions from input pictures, tailored explicitly for robotic painting tasks. Our pipeline addresses several key objectives, including the ability to paint from background to foreground, maintain fine details, capture structured regions accurately, and highlight important objects. To achieve this, we employ a panoptic segmentation network to predict the semantic class membership for each pixel in the image. This step provides us with a detailed understanding of the object categories present in the scene. Building upon the semantic segmentation results, we combine them with a color-based image over-segmentation technique. This process partitions the image into monochromatic regions, each corresponding to a specific semantic object. Next, we construct a hierarchical tree based on the segmentation results, which allows us to merge adjacent regions based on their color difference and semantic class. We take care to ensure that shapes belonging to different semantic objects are not merged together. We iteratively perform adjacency merging until no further combinations are possible, resulting in a refined hierarchical shape tree. To obtain the desired image abstraction, we filter the hierarchical shape tree by examining factors such as color differences, relative sizes, and the layering within the hierarchy of each region in relation to their parent regions. By employing this approach, we can preserve fine details, apply local filtering operations, and effectively combine regions with structured shapes. This results in image abstractions well-suited for robotic painting applications and artistic renderings.

CCS Concepts

• ***Computing methodologies*** → *Image processing; Image segmentation; Shape representations;* ***Non-photorealistic rendering;***

1. Introduction

The E-David system is a robotic painting platform enabling machines to create pictures using paintbrushes [GPD20]. One goal is to study human creativity by imitating the painting process as accurately as possible. Earlier attempts rendered images by overlaying semi-transparent and isolated brush strokes. However, human painters create interacting regions on the canvas, the replication of which is currently investigated.

The basis for such a region-based approach is a **Tree of Shapes (ToS)**, a data structure that represents the regions of an image in a hierarchical way [FXDG17]. A *region* is a set of connected pixels representing an image's feature. While the original approach contains tens of thousands of regions, we eliminate most by selectively merging nodes based on color differences and semantic classes. This yields trees of only dozens of regions which are realized by the painting robot in a background-to-foreground manner, allowing us to represent more important elements of a painting in a different style and with different tools. To achieve a compact and paintable Tree of Shapes, we require abstraction methods.

Most current frameworks for image abstraction [CLH*16, AMFM10, TJZ07, ASS*10] are based purely on accessible image information such as image edges and pixel colors. They lack information about image semantics and object relevance. In most cases, simple edge detection is not enough to provide reliable cues for distinguishing between image objects wherever the colors along the semantic borders are indistinct, or colors are locally very similar. Examples are grey cars on grey streets, concrete buildings next to the pavement, or gradual transitions from foliage to clouds.

With the proposed framework, we want to overcome this lack of information and address two key challenges for artistic image abstraction: First, the decision which details in the image need to be preserved, which can be removed, and how to vary this for different areas of the image. Second, an abstraction must handle indistinct borders between objects of the same or different importance. Abstraction methods without semantic information might merge these two objects, creating a region of the same color, spanning parts of both objects [CLH*16, AMFM10]. This behavior is often undesirable, as it creates a visual conflict for viewers since parts of distinct objects are represented by a shared region.

In our work, we propose using panoptic image segmentation [KHG*19] to obtain semantic information, which allows us to identify the types and pixel locations of scene elements. We combine

DOI: 10.2312/vmv.20231235

a **Felzenswalb-Huttenlocher (FZH)** color segmentation [FH04] as initial over-segmentation with the prediction of a panoptic segmentation model [WZA*21] that identifies semantic scene objects and their image borders to guide the abstraction strength locally for each detected semantic class and ensure that critical semantic borders remain present in the final abstraction.

To create a high-fidelity smoothing effect while losing as few image details as possible of the abstract regions, we introduce an iterative smoothing procedure based on our definition of local geometric coherence.

2. Related Work

Some methods for image abstraction employ a fixed abstraction rate or strength for the entire image, regarding all image features in the same manner. However, this results in either many small and unimportant shapes and regions remaining in the background or a very homogenous abstraction where the back- and foreground elements appear to have the same visual complexity [CLH*16, SAM17]. Other frameworks approximate different abstraction strengths, such as a painterly rendering approach that uses varying brush sizes [Her98] or multi-scale image filtering schemes [KSKD10] to give more detail to regions with more color changes and thus complexity. Some methods also employ user assistance to provide semantic image information to guide the abstraction, such as user scribbles [TGVB13] for foreground and background differentiation or eye-tracking systems [SD02].

Other region-based image abstraction methods use image segmentation or region growth methods. However, these methods only consider color information as discerning cues to derive region borders. Some methods over-segment the input and use clustering to aggregate similarly colored and adjacent segments [TJZ07, DMC15, ASS*10]. The Watershed [BM93], and quickshift [VS08] segmentation methods are based on changes of pixel color intensity and estimate strong edges as region borders. The FZH segmentation [FH04] produces segments, such that their final extent is defined along detected outlines of image elements. The algorithm starts by placing each image pixel into a separate component and then merges adjacent components based on the dissimilarity between border pixels and the inner dissimilarity of the components.

By assigning each region an appropriate aggregate color value, these image segmentation methods alone are often interpreted as image abstraction techniques or may serve as the basis for other abstraction methods [HEMK98, CLH*16, AMFM10].

Newer methods are based on neural networks. The **MaX-DeepLab** framework [WZA*21] is a transformer-based supervised machine learning framework for panoptic image segmentation. This segmentation task was first proposed and employed by [KHG*19]. It is defined as a coherent scene segmentation that labels each pixel to object classes and, if applicable, to class instances. Panoptic segmentation combines the two usually distinct tasks of semantic segmentation and instance segmentation. The author also sets the first baseline result in the field and defined a quality metric, panoptic quality (PQ), to compare panoptic segmentation methods.

While [KHG*19] followed a two-path strategy for creating semantic and instance segmentations separately and then combining these in the final output, [WZA*21] propose simplifying the panoptic pipeline to a transformer model that can predict a final class label immediately. They use a dual-path layout with a convolutional neural network (CNN) and a global memory to communicate between CNN layers.

The Tree of Shapes (ToS) was proposed by Faraj et al. [FXDG17], who describe it as a shape tree based on shape inclusion for shape abstraction. They create a topographic map based on the luminance in the HSV color space. The image is decomposed into shapes whose pixels share the same luminance. They then connect the shapes from high to low luminance to create a hierarchy between the shape while ensuring inclusion between parent and child nodes. Using this hierarchical structure, they propose that the shapes can then be simplified, replaced with primitive shapes, or filtered to create an abstraction after drawing the shapes in the hierarchy onto an image.

The result, however, often contains tens of thousands of nodes that need to be combined to create a useful base for robotic painting. Figure 1 shows the general abstraction pipeline as proposed by [FXDG17] from an input image to an abstraction where the shape tree was filtered and the shapes replaced with other similar shapes from a dictionary.

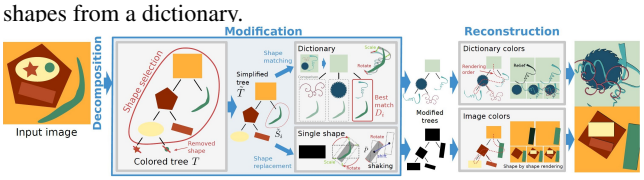


Figure 1: *Image abstraction using a ToS [FXDG17]*

3. Methodology

Besides creating an efficient ToS representation, our goal is to achieve a hierarchical abstraction incorporating semantic knowledge from machine learning models to guide the abstraction strength for individual object classes (and objects) in the input image. To this end, we combine a pre-trained model of the MaX-Deeplab [WZA*21] panoptic segmentation framework with a color-based over-segmentation that we obtain from the FZH algorithm [FH04] in a hierarchical shape tree that is based on a ToS. Our proposed method can work with any initial over-segmentation algorithm with minimal changes. However, other methods like SLIC superpixles tend to create very regularly shapes regions, especially within areas of little color difference. These regular structures can remain in the final abstraction for high detail areas and induce a mosaic like look, which we did not aim for with this framework. While it is possible to adjust the compactness to suit the input image this would introduce an additional hyperparameter.Figure 2 shows some example results of the FZH segmentation.

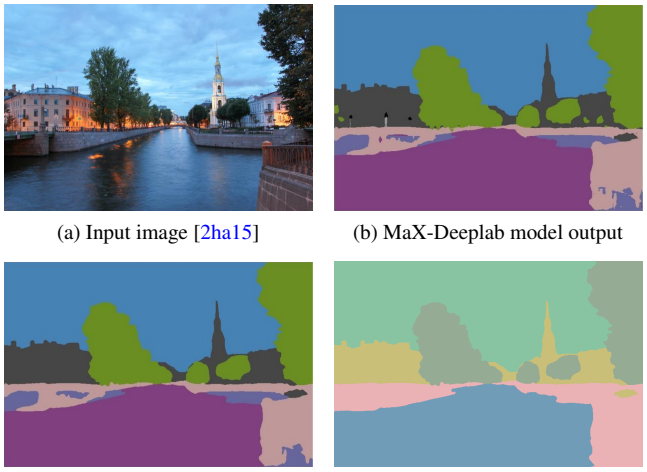
3.1. Panoptic Segmentation

We use a pre-trained model on the Cityscapes dataset [COR*16] of the MaX-Deeplab framework that creates a panoptic segmentation from an input image. It predicts an object class and corresponding instance whenever applicable for all input image pixels. Since

Figure 2: *Results of the FZH algorithm.*

the output of the MaX-Deeplab framework optimizes the prediction accuracy on the training set, it does not enforce the geometric coherence of present objects or judge how realistic the output regions are. As a consequence, the outputs can include many minor detection artifacts. These may include ragged edges along object borders or very small instances of objects which are only a few pixels wide. These might be misclassifications or small instances in the background of the image.

For our image abstraction, however, we desire large, coherent regions where the user can choose the abstraction strength. To meet these requirements and achieve smooth object borders, we remove all connected pixels of the same object with an area below a certain threshold. Then we reassign the pixels of these regions to the most dominant object class in their surroundings. As a final step, a user can select object classes to be merged together to remove unimportant object differentiations from the panoptic prediction. Figure 3 shows the prediction of the MaX-Deeplab framework and the results of the described processing.



(a) Input image [2ha15] (b) MaX-Deeplab model output

(c) Small artifact removal (d) Final panoptic segmentation

Figure 3: *Panoptic segmentation of an input image. (d) shows the panoptic map after user input to merge regions*

3.2. Merging Regions for the Tree of Shapes

We combine the smoothed semantic segmentation and the FZH segmentation to construct the first layer of the ToS, which contains the most detailed segmentation of the input and will serve as a starting point for the abstraction. We acquire an FZH segmentation of very small regions by applying a weak gaussian smoothing and a small value for the minimum region size. and split these along the object edges in the semantic segmentation. This ensures that every pixel of each resulting region is part of exactly one panoptic class object.

From here on, we merge adjacent regions to construct the hierarchical ToS. Every layer l added to the ToS represents a further abstraction. We obtain the shape adjacency for all regions on a specific layer by rendering all regions and constructing the region adjacency graph (RAG) from the rendered image. Using these adjacencies, we compute connected groups of regions we want to merge by checking our merge criteria for each node and its edges in the adjacency graph. This criteria for merging two adjacent regions on the same layer considers the two regions' relative sizes and color differences and previous merges involving the central node. It penalizes comparatively small and isolated regions by merging them earlier with larger and surrounding regions in the ToS hierarchy. However, the merging still primarily depends on the color difference of the shapes and is only added when the region size difference is below a minimum user-defined threshold $\alpha \in (0,1]$.

Due to its construction, the ToS consists of shape regions containing at least one pixel. Consider node $a \in RAG(l)$ and nodes $n,...,k,b \in RAG(l)$ that share an edge with it and are from the **same panoptic object**. Let $a_c, n_c, ..., k_c, b_c \in [0,255]^3$ be the regions' colors, and $a_s, n_s, ..., k_s, b_s \in \mathbb{N}_+$ the number of pixels in the regions. For the color difference as a similarity measure, we use ΔE, the CIE Delta E 2000 color difference [LCR01]. Furthermore, let $a_s \leq n_s \leq, ..., \leq k_s \leq b_s$ hold w.l.o.g. From which we can follow: $\frac{n_s}{a_s} \leq ... \leq \frac{k_s}{a_s} \leq \frac{b_s}{a_s}$. We choose p_s as the constant penalization weight for small regions. Then, for a threshold t, we merge regions a and b under the following conditions:

- If $\frac{a_s}{b_s} < \alpha$ and $(\deg(a) = 1$ or $\deg(b) = 1)$
- If $\frac{a_s}{b_s} < \alpha$ and $\deg(a) \neq 1$ and $\deg(b) \neq 1$ and $\Delta E(a_c, b_c) - t \cdot \left(-\log\left(\frac{a_s}{b_s \cdot p_s}\right)\right) \leq t$
- If $\frac{a_s}{b_s} \geq \alpha$ and $\Delta E(a_c, b_c) - t \cdot \left(-\log\left(\frac{a_s}{b_s}\right)\right) \leq t$
- $\sum_{\text{node} \in \{n_s,...,k_s,b_s\}} \Delta E(\text{node}_c, a_c) \leq t^2$

The function $\deg()$ yields the region's degree in the adjacency graph. We determined that a set of 22 thresholds can create a ToS representation that contains useful abstraction scales of the input. To select these thresholds, we did a stratified sampling of the semantic perception ranges described in Table 1, specific for our artistic abstraction application. Within the interval $[1,10]$ we sample 10 thresholds uniformly to create a good coverage of detailed abstractions within the first layers of the ToS. For values more similar than opposite colors within $(10,49]$, we sampled another 10 values uniformly. From the final strata we selected thresholds of 75 and 100 to ensure that also the highest color differences are covered and the final layer comprises all possible color merges. The used thresholds can be changed for specific input images but are usually fixed as described.

Table 1: *ΔE values according to [Sch20]*

Delta E	Perception
≤ 1	Not perceptible by human eyes.
1 - 2	Perceptible through close observation.
3 - 10	Perceptible at a glance.
11 - 49	Colors are more similar than opposite.
100	Colors are the exact opposite.

3.3. Filtering regions within the Tree of Shapes

After constructing the ToS, we have a hierarchical segmentation, for which we now need to determine what regions at what levels of details we want to keep in the final image abstraction. On the top layer of the ToS hierarchy are all objects and their disconnected subcomponents detected by the semantic segmentation at their maximum extent. Their subtrees contain all merged regions for the different merge thresholds within this panoptic class that are non-overlapping on the same layer. Users can assign an individual abstraction strength for all semantic classes and instances.

To create a proper image abstraction, we need to select a suitable subset of regions within the ToS by using good abstraction strength values for semantic objects to filter important ToS subtrees less, while removing most details from unimportant subtrees. For that, we can define the following filtering scheme and criterium for each parent and child pair in the ToS.

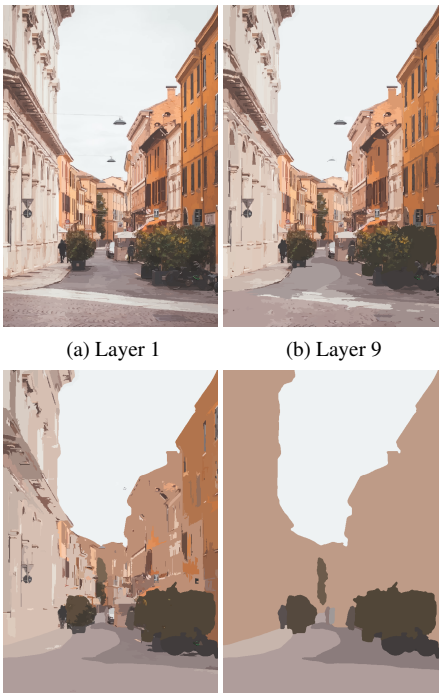
We traverse the nodes of the ToS in reversed breadth-first order. This ordering allows us to visit the nodes from the fine ToS shapes to the root node layer by layer. The filter checks the child's relevance for each node compared to its parent. It removes nodes that do not contribute enough new color information for their relative size compared to their parent under the user-specified abstraction strength parameter for the current subtree. We model region relevance and removal of nodes and their connected subtree by removing them if they satisfy the following condition:

$$\Delta E(a_c, b_c) + \Delta E(a_c, b_c) \cdot \frac{l_1}{l_{max}} < \gamma_p$$

Symbols a and b refer to the child and parent region and $l_1, l_{lmax} \in \mathbb{N}$ to child layer and maximal layer of the ToS, $\gamma_p \in [0, 100]$ is a user-defined hyperparameter that can be set individually for each semantic class. The subscript p refers to the semantic class to which child and parent belong. With this parameter, a user can set the minimum required color difference for a child node.

We selected the set of thresholds for constructing the ToS based on perceptual categories for a general context [Sch20] and added additional thresholds. Especially for minor color differences, we want to create more perceptually relevant steps in the hierarchy based on how many merges the thresholds produce. This selection of thresholds seems appropriate, as ΔE is defined only based on perceptual color differences but also depends on the available color space of the application and the context o f use.

Region filtering additionally incorporates the ToS layer quotient between the child region and the maximal reachable layer to model more important regions preserved during the merging. Providing a color difference bonus according to the shape layer should give additional weight to medium-sized or large regions with color differences slightly below the given user color thresholds. The user thresholds for the panoptic class regions are inspired by the Δ E color difference and share its typical range. Therefore, they should provide users with intuition on appropriate values for their desired setting based on the interpretations of perceived color difference values as presented in Table 1. Figure 4 shows some of the intermediate layers constructed by the ToS merging procedure.



(a) Layer 1 (b) Layer 9

(c) Layer 17 (d) Layer 23

Figure 4: *Illustration of the ToS structure. Layer 1 shows a minimally abstracted set of regions, and layer 23 has the highest abstraction with average colors. Original image [Zhy19]*

3.4. Local Region Processing

Regular mean filters are not applicable for our application, as they introduce mean values that do not correspond to existing regions. Instead, we require using only existing values for smoothing, so we need to use something akin to median filters. However, median filters smooth across visually significant semantic borders. We must prevent that, as we assume these borders contain a visually important distinction between two semantic objects.

Additionally, median filters might disrupt the geometry of important shapes where exact borders are needed. An example would be window frames of buildings. Finally, median filters often create fuzzy borders around the shape regions. These fuzzy region borders make the entire image look smoother than the original, but we lose important sharp edges in the result. The goal of the proposed smoothing is to remove as little detail from the shapes as possible while still creating coherent borders wherever possible.

To enforce smooth borders between individual region shapes, we introduce a notion of local coherence. Using the semantic information encoded in the ToS, we avoid processing across semantic borders in the following manner: we create rendered images of all shapes with the same panoptic class using their label as color. Next, we set all other pixels that are not part of any shape to a different value. These regions will be smoothed subsequently. By differentiating by class we can also control what objects are smoothed at all to preserve very precise structures.

We also use semantic knowledge in the later processing with robotic painting by processing semantic objects differently if we

need to preserve their local geometry. Such objects can be drawn with a small brush. An example use case would be to preserve text on road signs.

3.4.1. Structural Coherence

The local structural coherence is defined on a set of 3×3 neighborhoods (see Figure 5). We check whether the positions at the green entries all have the same label or are outside the current semantic object. If satisfied for any of these kernels, then the current center pixel is considered coherent to its region. Otherwise, we replace it with a label in the neighborhood that does. If there is no suitable label, we mark the pixel and continue. How to resolve these instances is discussed in the next section. If multiple candidate labels satisfy the structural coherence, we choose the label which has the closest mean color to the color the pixel has in the original image. This method can be seen as a semantics-based morphological filter.

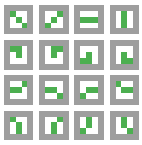
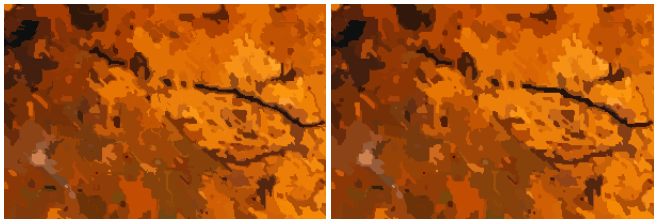


Figure 5: *Kernels*



(a) Filtered ToS rendering (b) Result after smoothing regions

(c) Reassigned pixels (in white)

Figure 6: *Results of the proposed region smoothing, the method acts like a semantics-based morphological filter.*

We used the Numba [LPS15] library to access all pixels and their neighborhoods in parallel for each iteration to increase the performance. We also compute the enforcement method for all semantic class renderings in parallel to increase performance.

3.4.2. Filling the Gaps

After we enforce the structural coherence of all pixels and regions, we may have pixels that cannot be assigned to any region without breaking coherence. To resolve this issue, we create new regions from connected groups of unassigned pixels. We overlap each connected group with the rendering of all shapes before enforcing coherence. We then create new regions using the intersection of this connected group with the regions in the rendering.

3.4.3. Reconstruction

Finally, update the geometry of all smoothed regions in the ToS and remove all regions that are no longer visible. Some regions may disappear as all their pixels were reassigned during smoothing. We do that by iterating over all available region labels across all rendered shapes, filtering them for this label, and creating a binary mask with all pixels assigned to this value. Lastly, we replace the available binary masks of the shapes with these extracted extents. Figure 6 shows the results of applying our proposed semantics-based morphological smoothing method on the filtered ToS.

4. Results

Figure 7 and Figure 8 show results from the proposed framework. In robotic painting, semantically unimportant regions will be represented by larger brushes and simple line patterns, and more important regions will be drawn with more precision and details.

We first create a panoptic mask for all images, with which we create a ToS. This ToS is then filtered, and all shapes are smoothed using the above method. The remaining shapes are then overlaid on an image to create the final abstraction. In the results, the sky is abstracted to one to two shapes, as these details were defined not to be important. Complex, semantically more important objects such as people, cars, or buildings are only slightly abstracted to create high-fidelity regions. Structured regions that are not so important for a scene, such as the vegetation and roads, were strongly abstracted as backgrounds. Our method gives much freedom to the user to define importance in terms of semantic object categories and instances.

In 8c, we can see the effect of our method. The street and sky have been selected for significant abstraction, leaving only one region representing the street. Similarly, foliage has been reduced in detail while preserving some color differences in 9b. Furthermore, due to semantic information, street signs and lampposts are preserved despite their proximity to foliage. Finally, the cars and people visible in the scene have been preserved with a high level of detail in both images, drawing the focus onto them as the main subject. This shows that our method can selectively abstract images like human painters.

Shape count is also significantly reduced in Figure 7. The initial segmentation contained 5423 regions which our method reduced to 1195. Significant reductions were made in the semantic classes "sky" (49 to 1 regions) and "road" (902 to 73 regions). In Figure 11, we achieved a reduction from 2925 to 840 regions while preserving the depicted people. The class "road" went from 2453 to 4 regions, while the two people were reduced from 1142 to 949 and 1953 to 1206 regions, respectively. The result images retain most of the relevant information, and the number of regions is sufficiently reduced for a painting robot.

Figure 7 again shows the improvement offered by semantic information. The sky and road are only represented by one region in the final output, while detail is preserved for cars, people, and building facades to a lesser degree.

Figure 11 illustrates the use for further artistic applications beyond abstraction. Using the semantic maps, we can direct a

painterly renderer to place finer strokes in regions of high importance while reducing the fidelity in less relevant regions. A painting robot can use the structural and geometric information acquired by our method to derive a motion plan for painting.

In Figure 9 we used an abstraction with a robotic painting system [GPD20] to get a painterly rendering of the proposed rendering pipeline to demonstrate that abstracted images can be used for robotic painting. In this setup the areas detected by our method are color quantized and then classified into areas, stroke groups and details. These are then realized by the e-David system using visual feedback.

(a) Original image [Pat18] (b) Result of the proposed framework

Figure 7: *An abstraction of a foilage-heavy scene: The overall leaf structure is summarized with a few shapes while preserving details like branches. Roadsigns and lamp posts are preserved despite covering a small area and having similar colors to some of their backgrounds.*

5. Discussion

For region merging and region filtering, we used custom heuristic criteria that model the regions' relevance depending on a set of general and user-defined thresholds specific to the input image. In addition, these criteria model the importance of region pairs for adjacent neighbors and child-/parent relationships in the ToS.

The conditions for merging two adjacent regions prioritize merging small and isolated regions. Medium-sized and larger regions are merged at their appropriate perceptual color differences. Penalizing smaller shapes is justified since smaller areas with perceptually similar colors to adjacent regions often only marginally add more detail to the image. Therefore, we apply a logarithmic penalty depending on the size quotient of the compared shapes and further penalize isolated regions with a degree of 1.

Large merge groups can occur when a central region has a sufficiently small color difference to each region, but the regions amongst themselves can have a much larger total color difference. We prevent such large groups from merging in a single layer by using the sum of previous merges as a limitation for further merges associated with the same node.

5.1. Region Smoothing

For specific images, applying regular median filters might introduce a set of new discontinuities between regions or even new artifacts to the image. Particularly where regions have delicate parts only a few pixels thick between larger subparts or along curved borders and can not guarantee that the final result would appropriately smooth all region borders. The smoothing mentioned above ensures we get regions that satisfy the defined coherence condition everywhere. However, our definition of local coherence removes some pixels and small regions, as they can not assign to other adjacent regions. These regions are the only ones that might still be incoherent after construction during the filling method. Therefore, we fill them in a way that keeps high visual fidelity to the original. Additionally we can also leverage the semantic knowledge to exclude specific objects from the filter if full fidelity is wanted locally.

5.2. Comparison of Smoothing Methods

Usually, image smoothing methods such as a gaussian, mean, or median filter smooth an image using a fixed-sized kernel. These filters smooth all image features regardless of their local structure and along all directions with the same strength. An issue here is that the shapes resulting from the proposed method may contain features that require different sizes of the smoothing kernel to be successfully removed from the image to create smooth region borders everywhere in the image. Instead of approximating such kernel size hyperparameters locally, we defined the geometric structural coherence as a smoothing paradigm. Figure 10 compares our method to a regular 3×3 kernel size median filter. The median filtering creates fuzzy borders, which can be avoided with the proposed method and preserves finer details in the image.

6. Conclusion and Limitations

We present a framework for guided, hierarchical image abstraction that combines image color- and semantic segmentation. The resulting regions are organized in a hierarchical Tree of Shapes with a hierarchy constructed using CIE Delta E color differences and relative shape size differences.

Our framework employs two new region merge and filter conditions based on colors and relative sizes. An image-smoothing procedure is used to reduce semantic noise artifacts and to smooth the abstracted regions. This smoothing method relies on geometric coherence for local neighborhoods as a paradigm to reassign pixels to different shapes. This way, we achieve region smoothing while preserving local region details.

With a single parameter for each panoptic class, the framework allows users to reduce the number of shapes representing a region they are uninterested in or keep the region complex with many different shapes. This abstraction is then further used in robotic painting to realize paintings in a back-to-front, background-to-foreground manner. The semantic information also allows us to represent important objects in different styles and tools.

(a) Original image [Bre17] (b) Panoptic segmentation result (c) Result of the proposed framework

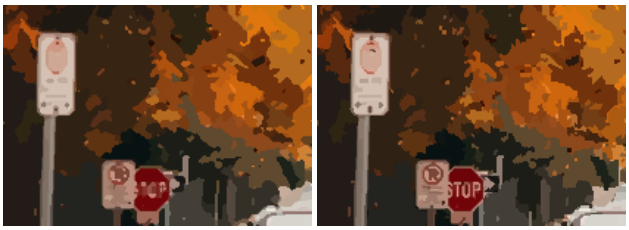
Figure 8: *An abstraction of a mostly constructed environment: People and cars are preserved at near full detail according to user input, while buildings are abstracted only slightly. The road is fully abstracted away.*

(a) Abstracted Landscape (our method) (b) Robot painted result

Figure 9: *Comparison between the results of our proposed abstraction method and using it in a robotic painting system [GPD20].*

Our method is limited by being dependent on the quality and accuracy of the panoptic segmentation. The segmentation provides object borders and thus determines exactly where two objects are split. This can be arbitrarily far away from the real object borders. Since the MaX-Deeplab panoptic segmentation framework we employ [WZA*21] is a supervised learning approach, the space of images we can use for our method is limited to what can be learned from the dataset, the model is trained on.

Large regions formed in a semantic class with high color variance get pushed towards grey colors in the ToS construction since



(a) Smoothed with 3×3 kernel median filter (b) Smoothed with the proposed method

Figure 10: *Comparison of a median filter and our proposed smoothing method*

we aggregate colors based on their mean. We slow down this process by ordering the merge steps based on the relative size and prevent merges of regions with large variances. However, large high-variance classes can still lead to artifacts in the final abstraction.

References

[2ha15] 2HAPPY:, 2015. [Online; accessed Jan. 20, 2022], Licensed under CC0 1.0. URL: https://www.stockvault.net/photo/175982/night-city-scene.

[AMFM10] ARBELAEZ P., MAIRE M., FOWLKES C., MALIK J.: Contour detection and hierarchical image segmentation. *IEEE trans. on Pattern Analysis and Machine Intelligence 33*, 5 (2010), 898–916. doi:10.1109/TPAMI.2010.161.

[And16] ANDRADE E.:, 2016. [Online; accessed Feb. 28, 2023], Unsplash License. URL: https://unsplash.com/de/fotos/Oo5sKlgIOZQ.

(a) Panoptic segmentation map (b) Painterly rendering (c) Described method (d) Panoptic segmentation map (e) Painterly rendering (f) Described method

Figure 11: *Two abstraction applications that use the semantic information obtained from panoptic segmentation. Simulated robotic painting method based on [Her98] using different brush sizes for fore- and background regions and our described method - original images [DS15] (first), and [And16] (second)*

[ASS*10] ACHANTA R., SHAJI A., SMITH K., LUCCHI A., FUA P., SÜSSTRUNK S.: *Slic superpixels*. Tech. rep., 2010.

[BM93] BEUCHER S., MEYER F.: The morphological approach to segmentation: the watershed transformation. In *Mathematical Morphology in Image Processing*. CRC Press, 2018 (1993), pp. 433–481.

[Bre17] BREVITÉ:, 2017. [Online; accessed Jan. 11, 2023], Unsplash License. URL: https://unsplash.com/es/fotos/gOHBaIl5WRc.

[CLH*16] CHENG M.-M., LIU Y., HOU Q., BIAN J., TORR P., HU S.-M., TU Z.: HFS: hierarchical feature selection for efficient image segmentation. In *Computer Vision – ECCV 2016* (2016), Springer Int. Publishing, pp. 867–882. doi:10.1007/978-3-319-46487-9_53.

[COR*16] CORDTS M., OMRAN M., RAMOS S., REHFELD T., ENZWEILER M., BENENSON R., FRANKE U., ROTH S., SCHIELE B.: The Cityscapes dataset for semantic urban scene understanding. In *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition* (2016), pp. 3213–3223.

[DMC15] DHANACHANDRA N., MANGLEM K., CHANU Y. J.: Image segmentation using K-means clustering algorithm and subtractive clustering algorithm. *Procedia Computer Science 54* (2015), 764–771. doi:10.1016/j.procs.2015.06.090.

[DS15] DE-SILVA S.:, 2015. [Online; accessed Feb. 28, 2023], Unsplash License. URL: https://unsplash.com/de/fotos/httxBNGKapo.

[FH04] FELZENSZWALB P. F., HUTTENLOCHER D. P.: Efficient graph-based image segmentation. *Int. Journ. of Computer Vision 59*, 2 (2004), 167–181. doi:10.1023/B:VISI.0000022288.19776.77.

[FXDG17] FARAJ N., XIA G.-S., DELON J., GOUSSEAU Y.: A generic framework for the structured abstraction of images. In *Expressive - NPAR 2017* (2017), NPAR '17 Proc. of the Symp. on Non-Photorealistic Animation and Rendering. doi:10.1145/3092919.3092930.

[GPD20] GÜLZOW J. M., PAETZOLD P., DEUSSEN O.: Recent developments regarding painting robots for research in automatic painting, artificial creativity, and machine learning. *Applied Sciences 10*, 10 (2020), 3396.

[HEMK98] HARIS K., EFSTRATIADIS S. N., MAGLAVERAS N., KATSAGGELOS A. K.: Hybrid image segmentation using watersheds and fast region merging. *IEEE Trans. on Image Processing 7*, 12 (1998), 1684–1699. doi:10.1109/83.730380.

[Her98] HERTZMANN A.: Painterly rendering with curved brush strokes of multiple sizes. In *Proc. of the 25th Ann. Conf. on Computer Graphics and Interactive Techniques* (1998), ACM, p. 453–460. doi:10.1145/280814.280951.

[KHG*19] KIRILLOV A., HE K., GIRSHICK R., ROTHER C., DOLLAR P.: Panoptic segmentation. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)* (2019).

[KSKD10] KYPRIANIDIS J. E., SEMMO A., KANG H., DÖLLNER J.: Anisotropic kuwahara filtering with polynomial weighting functions. In *TPCG* (2010), Eurographics Association, pp. 25–30.

[LCR01] LUO M. R., CUI G., RIGG B.: The development of the cie 2000 colour-difference formula: Ciede2000. *Color Research & Application 26*, 5 (2001), 340–350. doi:https://doi.org/10.1002/col.1049.

[LPS15] LAM S. K., PITROU A., SEIBERT S.: Numba: a llvm-based python jit compiler, 2015.

[Pat18] PATEL R.:, 2018. [Online; accessed Jan. 11, 2023], Unsplash License. URL: https://unsplash.com/es/fotos/yK_mJlzLQUY.

[SAM17] SADREAZAMI H., ASIF A., MOHAMMADI A.: Iterative graph-based filtering for image abstraction and stylization. *IEEE Transactions on Circuits and Systems II: Express Briefs 65*, 2 (2017), 251–255. doi:10.1109/TCSII.2017.2669866.

[Sch20] SCHUESSLER Z.: Delta E 101, 2020. Online; accessed December 20, 2022. URL: http://zschuessler.github.io/DeltaE/learn/.

[SD02] SANTELLA A., DECARLO D.: Abstracted painterly renderings using eye-tracking data. In *Proc of the 2nd Int. Symp. on Non-Photorealistic Animation and Rendering* (2002), ACM, p. 75–ff. doi:10.1145/508530.508544.

[TGVB13] TANG M., GORELICK L., VEKSLER O., BOYKOV Y.: GrabCut in one cut. In *Proc. of the IEEE Int. Conf. on Computer Vision (ICCV)* (2013), pp. 1769–1776.

[TJZ07] TAO W., JIN H., ZHANG Y.: Color image segmentation based on mean shift and normalized cuts. *IEEE Trans. on Systems, Man, and Cybernetics, Part B (Cybernetics) 37*, 5 (2007), 1382–1389. doi:10.1109/TSMCB.2007.902249.

[VS08] VEDALDI A., SOATTO S.: Quick shift and kernel methods for mode seeking. In *Computer Vision – ECCV 2008* (2008), Springer Berlin Heidelberg, pp. 705–718. doi:10.1007/978-3-540-88693-8_52.

[WZA*21] WANG H., ZHU Y., ADAM H., YUILLE A., CHEN L.-C.: MaX-DeepLab: End-to-End panoptic segmentation with mask transformers. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)* (2021), pp. 5463–5474.

[Zhy19] ZHYVCHIK E.:, 2019. [Online; accessed Dec. 24, 2022], Unsplash License. URL: https://unsplash.com/photos/06mwaWJ3HfM.