Iterative Image Saliency Segmentation and Inpainting: A Method for Unsupervised Multi-Region Segmentation

Jason Wan, *UCLA,* December 15th, 2023

Abstract

Unsupervised image saliency segmentation methods, segment the image into a background and salient foreground, however, in the real world, an image can contain multiple objects. This research presents an approach to extend image saliency segmentation to more than two regions using and leveraging inpainting; specifically, saliency segmentation with TokenCut and inpainting with Controlnet V1.1 Nightly and Deepfillv2. The core idea revolves around iteratively segmenting an image to identify areas with the lowest mutual information and subsequently inpainting these regions. The iterative process continues until no significant segment masks are detectable. Preliminary results show a notable improvement between Controlnet and Deepfill in segmenting intricate image details, showcasing the potential of this combined approach in various applications.

Code: https://github.com/nJasow04/deepfillv2-pytorch

Introduction

The primary objective was to develop a model that iteratively uses TokenCut to segment areas of an image with the lowest mutual information, followed by inpainting these areas with ControlNet-Nightly-V1-1. The process repeats until there are no significant segments to isolate. However, complications with ControlNet led to the second objective: employing DeepFillv2, which showed a marked improvement.

Image segmentation and inpainting are crucial in computer vision, with diverse applications from medical imaging to digital content creation. Traditional methods often struggle to accurately identify complex image regions. This research innovatively combines advanced techniques: TokenCut for image segmentation and ControlNet-Nightly-V1-1 and DeepFill for inpainting.

Saliency segmentation and inpainting, pivotal in this research, can be trained without human supervision. Typically, if an image contains a single foreground object, saliency segmentation algorithms can identify it autonomously. However, real-world scenarios often feature multiple objects. A natural approach to sequentially detect multiple objects involves identifying the most salient object, removing it from the scene, and then searching for the next. Since physically cutting an object is useless, we employ inpainting as a virtual removal tool. This project presents preliminary results from testing this methodology, exploring the synergy between TokenCut for segmentation and ControlNet/DeepFill for inpainting, effectively paving the way for a method of multiple object segmentation.

Methodology

3.1 Initial Setup and Environment Configuration

The foundation of this research involved cloning the necessary GitHub repositories for TokenCut and ControlNet. Given the distinct PyTorch version requirements for each, the initial phase focused on configuring and tweaking two separate environments to ensure compatibility and seamless integration.

3.2 Integration of Core Models

The key step was the integration of get_saliency.py from TokenCut's unsupervised_saliency_detector directory with the gradio_inpaint.py model from ControlNet. This integration was critical in creating a cohesive process where saliency detection and inpainting could be performed in an iterative loop. The bilateral solver within TokenCut was also utilized in this step.

3.3 Hardware Constraints and Solutions

Initially, the project utilized Vision Lab's Thel computer. However, the substantial GPU memory requirements of the gradio_inpaint model necessitated a switch to a more powerful machine, Zapp. This shift was essential to handle the intensive computational demands of the models.

3.4 Development of Visionloop.py

The integrated model, named Visionloop.py, was a novel creation specifically for this project. It incorporated the functionalities of both TokenCut and ControlNet, thereby enabling the iterative process of segmentation and inpainting.

3.5 Setting Text Prompts for ControlNet

Within Visionloop.py, specific text prompts were set for ControlNet, including "No Creativity," "Zero Imagination," "Clear," "Transparent Object," and a no-prompt condition. These prompts guided the inpainting process in ControlNet, influencing the nature of the inpainted segments.

3.6 Iterative Segmentation and Inpainting

The core of the methodology involved running TokenCut's segmentation algorithm to identify the segment with the least mutual information. Due to limitations in TokenCut’s masking capability, a dilation process was applied to the masks to ensure full coverage.

The dilated masks, along with the original image, were then input into ControlNet's gradio_inpaint function. This resulted in a new image where the selected segment was inpainted based on the provided text prompt. The output included the new image, the masks, and the image with the mask overlay, all of which were documented.

3.7 Iteration and Documentation

The path to the new image was stored for subsequent iterations. The process was designed to repeat for a maximum of five iterations or until TokenCut could no longer identify significant masks.

3.8 Comparison with DeepFillv2

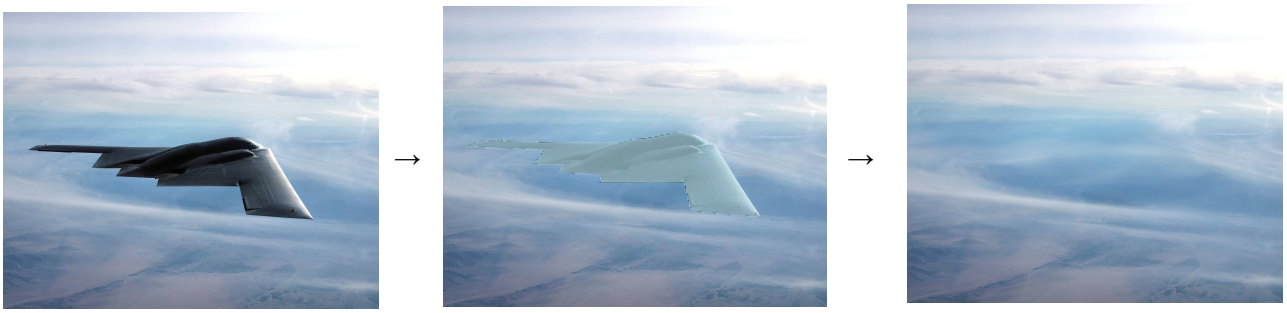
As a comparative study, the same iterative process was later replicated using DeepFillv2. The integration involved stitching DeepFillv2's test.py example code, following a similar workflow to assess and contrast the performance with the initial model.

Results

4.1 Initial Results with ControlNet and TokenCut

The first set of experiments involved using ControlNet with the prompt "No creativity" in combination with TokenCut for mask generation. The initial results were promising, the masks generated by TokenCut, after some dilation, provided decent segmentation. The subsequent inpainting using ControlNet on the first iteration resulted in acceptable quality.

Example 1: High Contrast image, easy mask



Plane with stark contrast with the sky → Dilated Mask → Inpainted result with clear sky

Example 2: More complex, blurry background

Camera → Dilated Mask → Inpainted result with reasonable hallucinated fingers

Example 3: Multi-Object Living Room

Room → Dilated Mask → Inpainted result with creative hallucinations

4.2 Issues with Subsequent Iterations

Despite the initial success, the model's performance deteriorated in subsequent iterations. The quality of results from ControlNet declined from the second iteration onward, showing an increasing trend of hallucinated and unrealistic outputs.

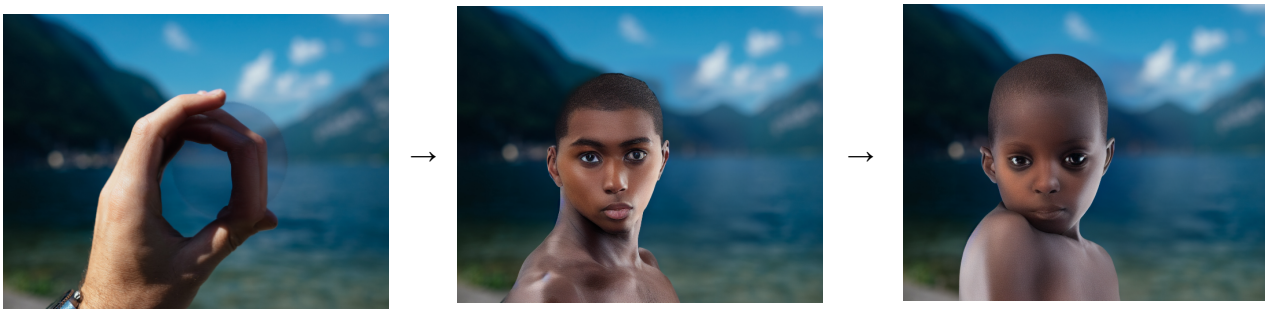
Example 1: Black Plane in the Sky (Controlnet Prompt: “Zero Creativity”)

First Iteration

Second Iteration:
Hallucinates a finger or arm

Third Iteration:
Hallucinates the actual
prompt “zero creativity”

Example 2: Camera (Controlnet Prompt: “Blend In”)



First Iteration

Second Iteration:
Extreme Hallucination

Third Iteration:
Continued Hallucination

Persistence of Hallucinations: Altering the text prompts in further tests did not mitigate the issue of hallucinations, indicating a limitation in ControlNet's ability to blend seamlessly with varying backgrounds.

4.3 Improved Results with DeepFillv2

Shifting to DeepFillv2 yielded better and more reasonable outcomes:

Example 1: Camera

First Iteration

→

First Segmentation

→

First Inpaint

→

Second Segmentation

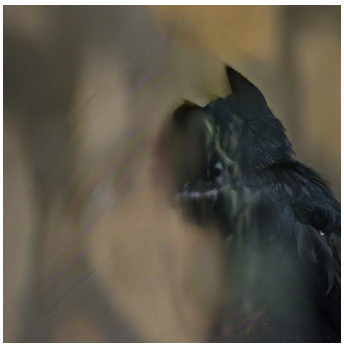
→

Final Image

Example 2: Bird

First Iteration

First Segmentation



First Inpaint

Second Segmentation

Second Inpaint

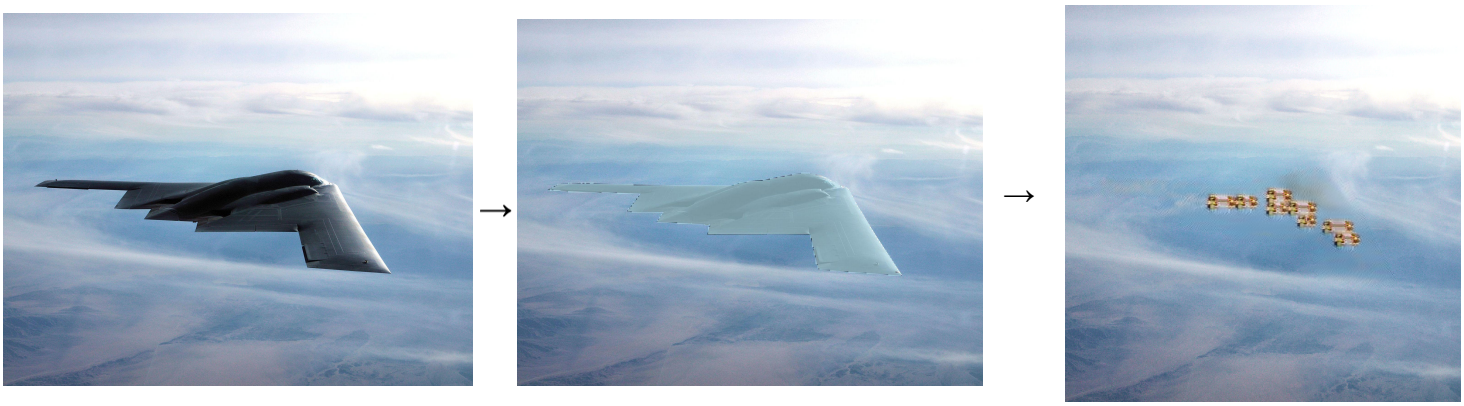
Third Segmentation

Final Image

4.4 Limitations and Future Directions

Hallucinations of Deepfillv2: Despite the improved results with DeepFillv2, the model may still hallucinate random artistic features reflecting the nature of its training as shown in the following example:

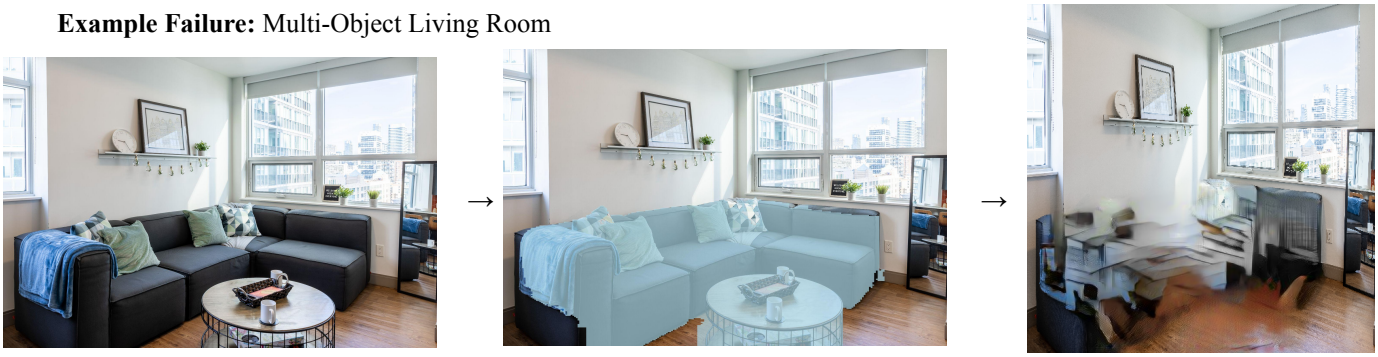
Example Failure: Black Plane in the Sky



Original Image | TokenCut Mask | Result: Strange painting-like Hallucinations

Limitation of TokenCut: The overall effectiveness is still constrained by TokenCut's limitations, particularly in handling complex images since TokenCut is based on Dino features, which are trained on ImageNet. When an image is too complex, TokenCut will simply fail to run or fail to correctly segment.

Example Failure: Multi-Object Living Room



Original Image | First Segmentation | First Iteration

Second Segmentation | Final Image

Prospective Use of Better Segmentation Devices: Future research will explore the implementation of Meta's SAM model for stronger segmentation capabilities, potentially overcoming the current limitations of TokenCut.

Conclusion

This research aimed to enhance image segmentation and inpainting using TokenCut and ControlNet, and later DeepFillv2. Initial experiments with ControlNet yielded promising results in the first iteration but showed a decline in performance in subsequent iterations, leading to unrealistic outputs. The subsequent use of DeepFillv2 instead of ControlNet improved the results, especially in the early iterations. However, the effectiveness was still limited by TokenCut's segmentation capabilities. These findings suggest that while the integration of segmentation and inpainting models can be effective, the choice of models is crucial, and limitations in segmentation capabilities can be a significant constraint. This research demonstrated the feasibility of automated segmentation and inpainting in scences with multiple objects, offering insights for future enhancements in image processing techniques.

References

Caron, M., Touvron, H., Misra, I., Jégou, H., Mairal, J., Bojanowski, P., & Joulin, A. (2021). Emerging Properties in Self-Supervised Vision Transformers. In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) (pp. 9650-9660).

Ilyasviel. (2023). ControlNet-v1-1-nightly [GitHub repository]. Retrieved October 30, 2023, from https://github.com/lllyasviel/ControlNet-v1-1-nightly

Kirillov, A., Mintun, E., Ravi, N., Mao, H., Rolland, C., Gustafson, L., Xiao, T., Whitehead, S., Berg, A. C., Lo, W.-Y., Dollár, P., & Girshick, R. (2023). Segment Anything.

Nipponjo. (2023). Deepfillv2-pytorch [GitHub repository]. Retrieved December 2, 2023, from https://github.com/nipponjo/deepfillv2-pytorch

Wang, Y. (2023). TokenCut [GitHub repository]. Retrieved October 13, 2023, from https://github.com/YangtaoWANG95/TokenCut

Wang, Y., Zhang, W., Wang, L., Liu, T., & Lu, H. (2022). Multi-Source Uncertainty Mining for Deep Unsupervised Saliency Detection. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (pp. 11727-11736).