Tracing Algorithm Improvements (11.03.2026)

Identified issues

1. When dealing with AI generated textures, some color artifacts may appear, producing very small islands that should otherwise be of the surrounding color
2. Due to the way the texture is generated, islands may extends out of the UV mapped area, rendering simple 3D projection impossible
3. The default color quantization method (MEDIANCUT) is heavily biased by the primary color, which “floods” the other colors
4. When a particular pattern covers multiple UV islands (i.e. crosses a UV seam), multiple traces will be produced, splitting the shape at the seam
5. The path produced by the tracer do not take into account the reachability of the zones, and will produce undrawable traces (e.g. edges of the eyes)
6. Fill slicing sometimes fail, inverting inside and outside areas of color islands

Improvements

The small artefact problem

Explanation

Some small color artifacts appear as isolated group of pixels within larger color areas.

- These arise when using the AI generated texture that presents less crisps color transition than how the `cv2.findContours` will handle.
- Also,texture image file format has an impact. For example the `.jpg` (doing more compression) present less quality color changes than `.png`.

Solution

Use a minimum surface threshold. Islands whose contour area falls below `min_island_surface` are directly discarded. Also, use .png texture format to maximise color quality.

Due to the hierachical recuperation of the contours, this has the side-effect that discarded coutours will be “automatically” merged into the surrounding one, independently of their respective colors.

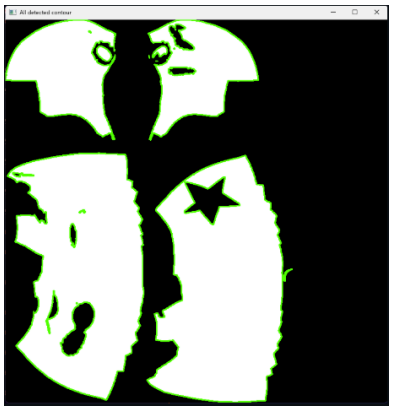

Figure 1: The green layer before change

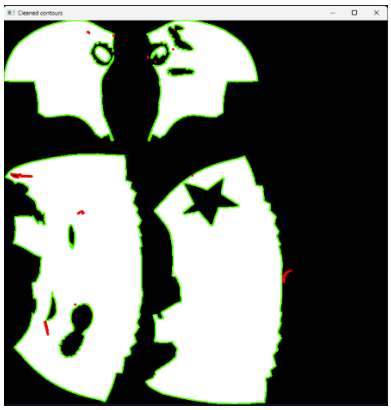

Figure 2: After the small contours detections

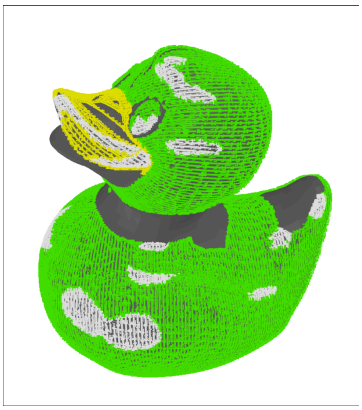

Figure 3: Thoses detected small contours are succesfully filled in green

A word on other tested approach

Using blur had been initally thought as a plausible solution as it's dilation effect could potentially "fill the holes". Tests and visual judgement lead to understand that it didn't fill the holes nor preserved the pattern boundaries. Thus blur approach was rejected.

Conclusion

This observed effects of the implemented described solution solve the issue stated. Based on multiple tested texture and palettes and visual comparison[1], I therefore conclude to this approach's correctness.

The "Out of bounds" problem

Possible solutions to the islands extending out of the UV map are:

- compute the intersection between UV geometries and island borders
- extrapolate projection outside of UV triangles
- apply a shrinked mask to only get polygons inside the UV map or shrink the polygons after contour detection

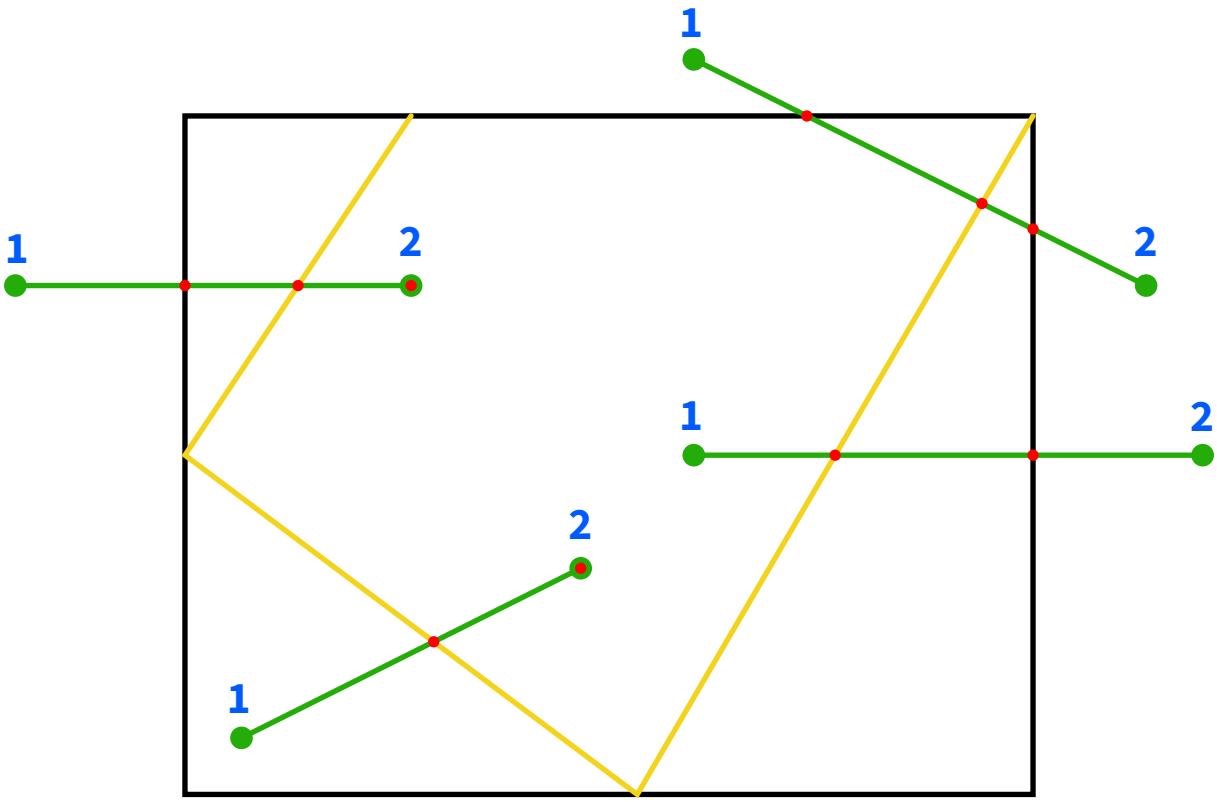

Figure 4: UV boundary crossing cases

The color quantization (palettization) problem

Explanation

The default color quantization method (MEDIANCUT) is heavily biased by the primary color, which "floods" the other colors.

This bias is inherent of the method — as a bucket-based algorithm, it splits color space at the statistical median of pixel count, allocating palette entries proportionally to color frequency.

[1] PR 74

In our pipeline this translate in the black color, beeing the one replacing elements not in the UV mask applied on the texture, thus having heavy weight.

Solution

Using a KNN-based approach[2] solves this issue[3]: instead of building the palette from pixel frequency, the model is trained on a fixed, predefined palette and assigns each pixel to its nearest color in that space — regardless of how dominant that color is in the image. This eliminates the frequency bias introduced by MEDIANCUT. This approach also allows specific colors (e.g. the background black) to be explicitly excluded from the palettization process.

Figure 5: Before the palettization update

Figure 6: After the palettization update

Conclusion

Using this update and adding the mask above the texture before palettization create a cleaned palettized texture with better color fidelity. A further add could be to allow multiple colors from the palette not to be draw. (For example, the yellow, as it is the base color of the 3d printed duck)

UV seams

To avoid splitting contours at UV seams, there could be a post-processing step which links polygons with overlapping (i.e. very close) points

Unreachable areas

A mask could be defined to limit drawing to manually selected “drawable” areas

Fill slicing

Explanation

Fill slicing raise exceptions and produce incorrect results on specific contours:

- `ValueError: A linearring requires at least 4 coordinates` — caused by contours with fewer than 4 points
- Fill slicing inverted, thus filling outside the shape. This is caused by groups of 3 consecutive colinear points in the contour, making the `.intersect()` method to fail.

Solution

- Contours with fewer than 4 points are rejected before geometry construction.
- Add a `clean_island()` function to removes the middle point of any colinear triplet. The colinerity is tesed with the cross-product.

[2] StackOverflow KNN solution
[3] PR 65

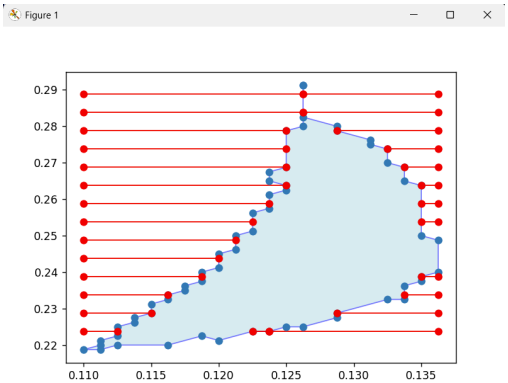

Figure 7: Before the fill slicing debug

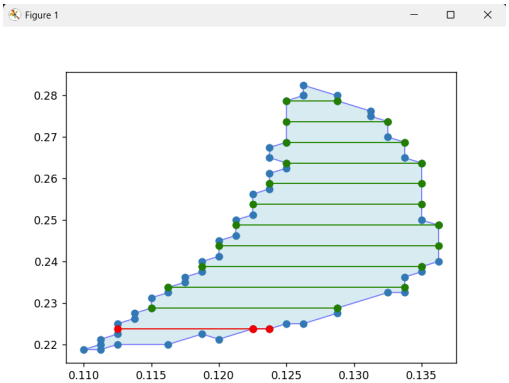

Figure 8: After the fill slicing debug

Conclusion

The observed effects of the implemented described solution solve the issue stated[4]:. Based on multiple tested texture and palettes and visual comparison, I therefore conclude to this approach's correctness.

—

Attack plan

Louis

- [◧] Out of bounds
- [?] UV seams
- [?] Unreachable areas

Jeremy

- [x] Finalize color quantization
- [x] Small artefacts replacement
- [x] Identify and solve fill slicing issues

—

Self- Reflection

Is this planned solution/the current pipeline orientation a good one ?

Yes	No
It works for simple textures	It breaks for more complex textures (currently)
Current issues seem to be solvable without complete refactoring	Some issues (e.g. UV seams) seem complicated to solve completely / elegantly

In conclusion we believe this is still a viable solution.

Jeremy Duc, Louis Heredero

[4] PR 74