Reduction of number of points in contours (23.03.2026)

Objective

Based on feedback from the team robot, my goal was to reduce the total amount of traces, thus the number of points in contours sent to the robot. In discussion with Pierre-Yves, we postulate that fewer points would mean easier, faster, and more reliable (less calibrations) robot paths. This work is recorded in the PR 92[1])

Improvements

N.1 - contours simplification

Explanation

After contour detection with the `cv2.findContours`, the raw contours contain a lot of points, often due to small direction changes that aren't significantly changing the global form of the detected shape. This number could be reduce with a contour approximation.

Solution

Use `cv2.approxPolyDP()` to each contour during the island detection phase. The parameter `contour_simplification_epsilon` (added to `config.py`) controls the tolerance of the point reduction. It act as a trade-off between fidelity and reduction.

A debug visualisation has also been made to visually observe the effect of simplification.

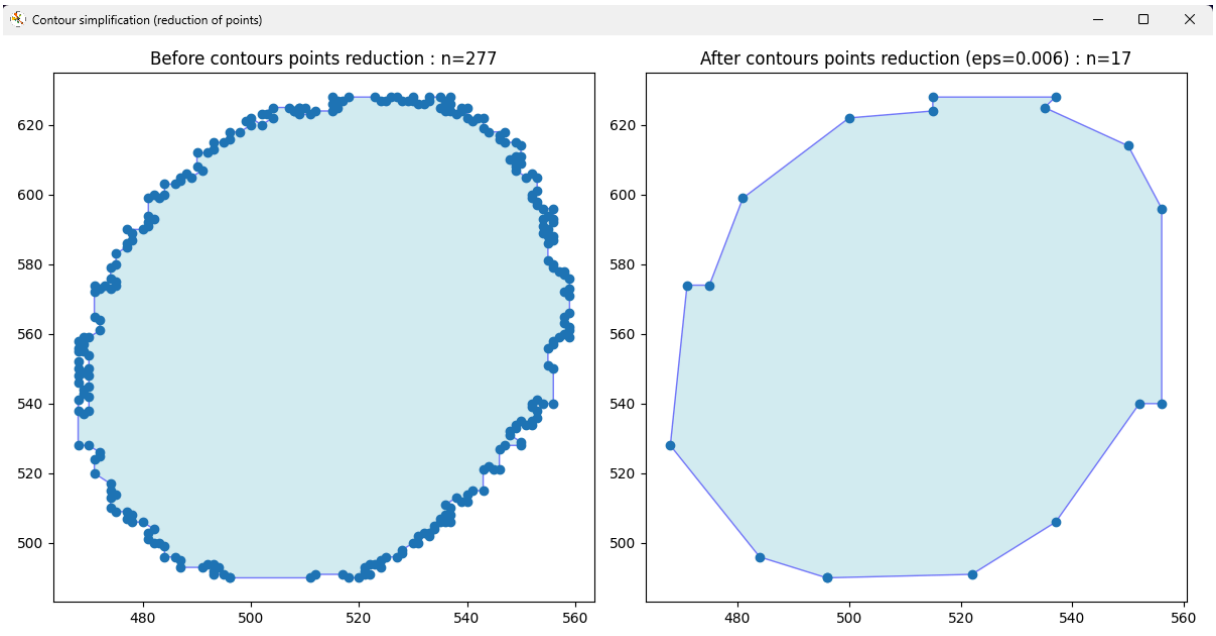

Figure 1: The debug visualisation

Conclusion

The process does reduce the number of point count on the contour polygons passed into the tracing pipeline. In example shown above, it went from a contour made of 277 to 17 points.

—

[1] PR 92 reduce number of points in contours

N.2 — Add an angular threshold between consecutive faces in trace projection

Explanation

During 3D trace projection, intermediate points are added at every face boundary crossed along the traced path. When two consecutive faces are almost coplanar (threshold), the extra intermediate point could be bypassed to reduce the total points amount.

Solution

Add an radian angular threshold `parallel_angle` (in config). Thus, before inserting an intermediate point, the angle between the normals of the two faces is checked. If the angle is under the threshold, the two faces are considered “too parallel” and no intermediate point is added.

A augmentation of the `min_segment_length` parameter was also done to avoid the insertion of short intermediate segment.

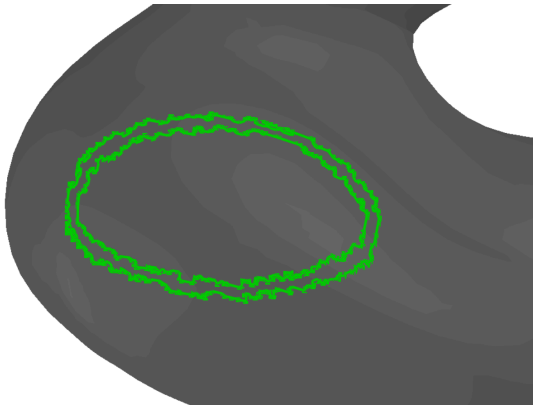

Figure 2: Before the trace reduction

Figure 3: After the trace reduction

Conclusion

After this change has been implemented, a the above example proved a reduction from **664 points** down to **139 points** to be drawn by the robot. Thus changes produce the desired goal.

—

Self-reflection

After discussion with the CTO and team robot members, I reflect on this change :

- It succesfully meets the target of reducing the number of points per traces
- The contour reduction sometimes simplify “too much” the shape, up to the point were it can almost loose it’s initial representation -> resampling the reduced contour using spline is potential update
- Some diffrent colors traces overlaps on the model due to contour simplification
- I should (and went) go to the lab with the robot team to observe the effects of the changes -> the thickness of the pen should be imitated in the tracing process

Jeremy Duc