

Sensor Characterization and Line Fitting for Robot Localization

Heictor Alves de Oliveira Costa

November 28, 2024

Introduction

This report documents the implementation and testing of the `fitLine.m` function as part of the activity on sensor characterization and line fitting for robot localization. The activity focuses on extracting line features from laser scan data using the Split-and-Merge algorithm and performing line fitting to minimize total least squares errors.

The task involves calculating line parameters, such as the angle α and the perpendicular distance r , using a set of 2D points in Cartesian coordinates.

Task Description

The `fitLine.m` function was designed to:

- Compute the centroid of the input points.
- Calculate the orientation α of the line using the total least squares method.
- Determine the perpendicular distance r from the origin to the line.
- Handle negative radii by adjusting the parameters.

The task was implemented in MATLAB, and the results were validated using the provided test scripts.

Implementation

The implementation of the `fitLine.m` function followed these steps:

1. The centroid (x_c, y_c) of the input points was calculated using the arithmetic mean of the x and y coordinates.
2. The parameter α was derived using the formula:

$$\alpha = \frac{1}{2} \cdot \arctan 2 \left(-2 \sum (x_i - x_c)(y_i - y_c), \sum (y_i - y_c)^2 - \sum (x_i - x_c)^2 \right)$$

3. The parameter r was computed as:

$$r = x_c \cos(\alpha) + y_c \sin(\alpha)$$

4. Negative radii were eliminated by adjusting α and r .

The complete MATLAB code is included in the Appendix.

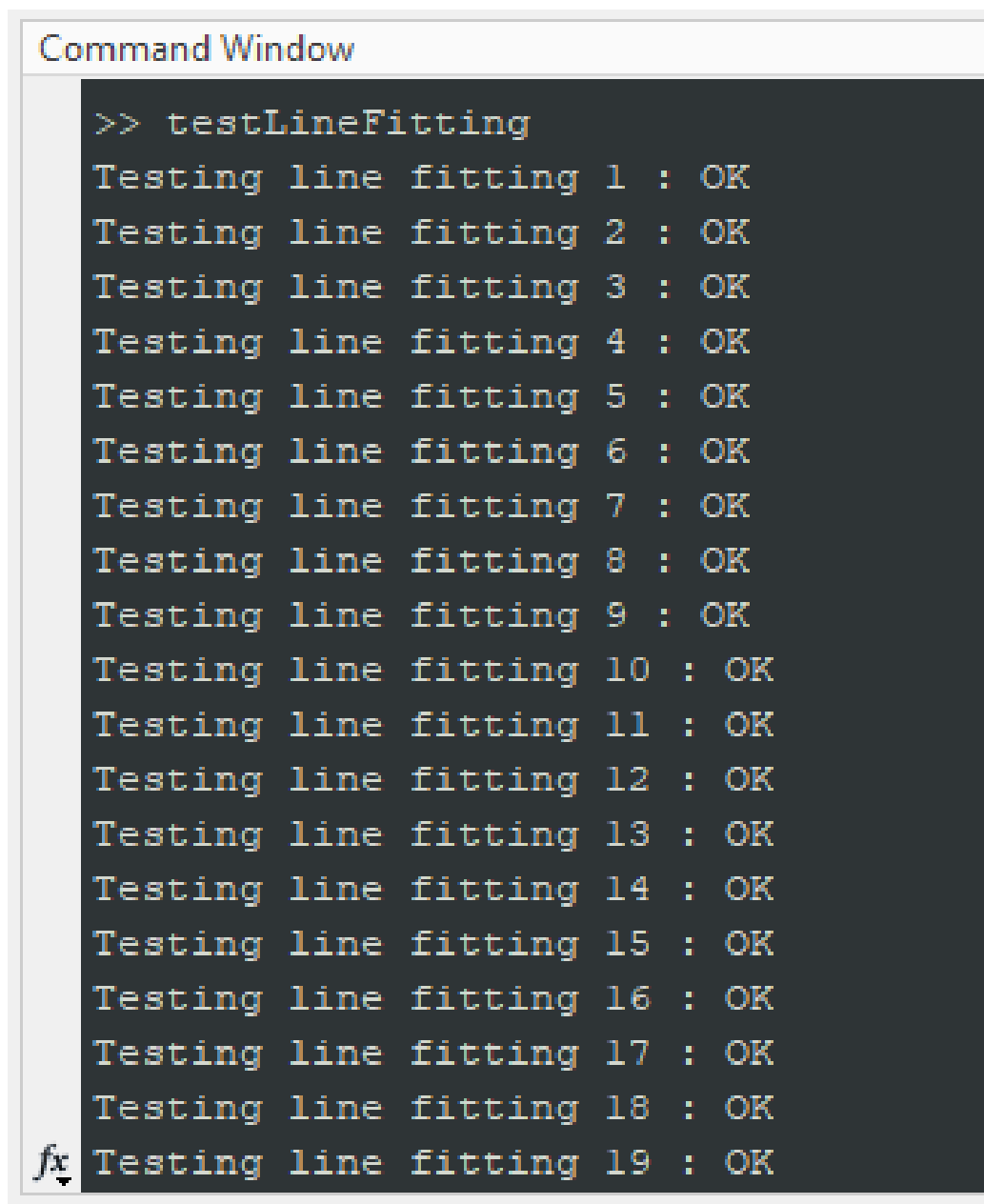
Testing

The implementation was tested using the provided scripts:

1. **testLineFitting.m:** This script tested the `fitLine.m` function on artificial data. The output indicated that all cases of "Testing laser scan" resulted in "OK."
2. **testLineExtraction.m:** This script validated the integration of `fitLine.m` within the Split-and-Merge algorithm. All cases of "Testing laser scan" resulted in "OK."

Results

The results confirmed the correctness of the `fitLine.m` function and its integration into the line extraction algorithm. Figures displaying the fitted lines and extracted line segments are included below.

A screenshot of a MATLAB Command Window. The title bar reads "Command Window". The window has a dark background with light-colored text. The text shows a series of 19 test results, each starting with ">> testLineFitting" followed by "Testing line fitting [number] : OK". The numbers range from 1 to 19. At the bottom left of the window, there is a small icon of a function handle 'fx' with a downward arrow.

```
>> testLineFitting
Testing line fitting 1 : OK
Testing line fitting 2 : OK
Testing line fitting 3 : OK
Testing line fitting 4 : OK
Testing line fitting 5 : OK
Testing line fitting 6 : OK
Testing line fitting 7 : OK
Testing line fitting 8 : OK
Testing line fitting 9 : OK
Testing line fitting 10 : OK
Testing line fitting 11 : OK
Testing line fitting 12 : OK
Testing line fitting 13 : OK
Testing line fitting 14 : OK
Testing line fitting 15 : OK
Testing line fitting 16 : OK
Testing line fitting 17 : OK
Testing line fitting 18 : OK
fx Testing line fitting 19 : OK
```

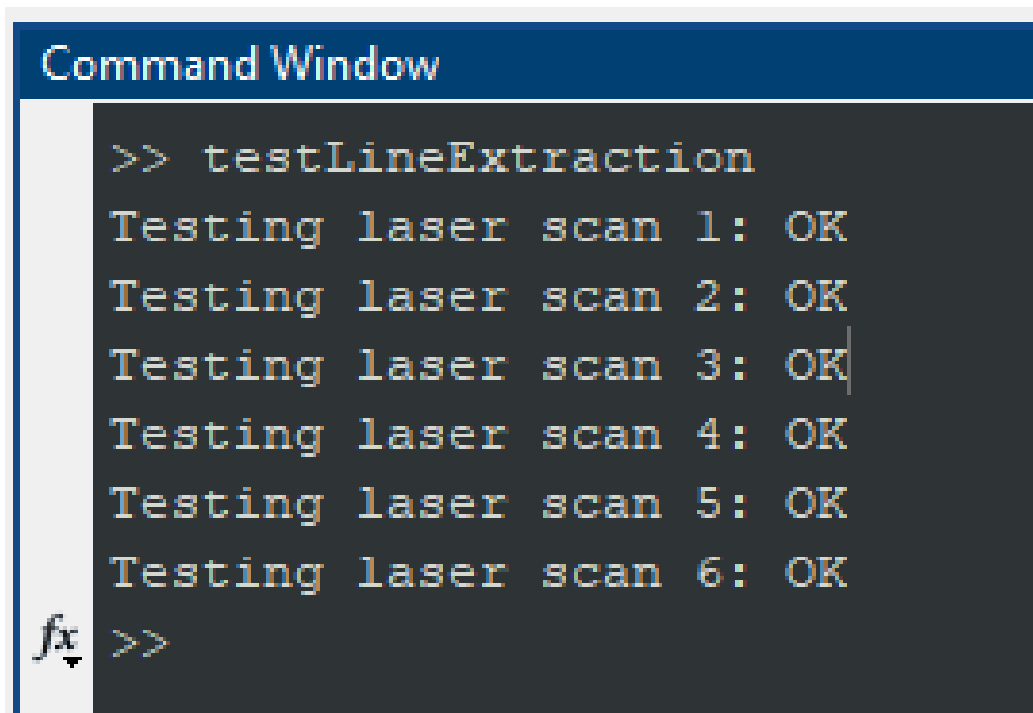
Figure 1: Line fitting results showing measured points and fitted lines.

Figure 1: Results of testLineFitting.m

Figure 2: Results of testLineExtraction.m

Conclusion

The fitLine.m function was successfully implemented and validated. The results demonstrated that the function³ correctly computes the parameters

A screenshot of a MATLAB Command Window. The title bar is dark blue with the text "Command Window" in white. The background is dark gray. The text is displayed in a monospaced font with syntax highlighting: ">> testLineExtraction" is in blue, and the subsequent lines "Testing laser scan 1: OK" through "Testing laser scan 6: OK" are in yellow. A cursor is visible at the end of the sixth line. At the bottom left, there is a small icon of a cursor with the letter 'f' and a subscript 'x' next to it, followed by ">>".

```
>> testLineExtraction
Testing laser scan 1: OK
Testing laser scan 2: OK
Testing laser scan 3: OK
Testing laser scan 4: OK
Testing laser scan 5: OK
Testing laser scan 6: OK
fx >>
```

Figure 2: Split-and-Merge results showing measured points and extracted line segments.

of a line while minimizing total least squares errors. The integration of this function into the Split-and-Merge algorithm also performed as expected, as evidenced by the test results.

Future work may involve optimizing the implementation for real-time applications and testing the algorithm on larger datasets.

Appendix: MATLAB Code for `fitLine.m`

```
function [alpha, r] = fitLine(XY)
% Compute the centroid of the point set (xmw, ymw)
    xc = mean(XY(1, :));
    yc = mean(XY(2, :));

% Compute parameter alpha
    nom = -2 * sum((XY(1, :) - xc) .* (XY(2, :) - yc));
    denom = sum((XY(2, :) - yc).^2) - sum((XY(1, :) - xc).^2);
    alpha = 0.5 * atan2(nom, denom);
```

```
% Compute parameter r
    r = xc * cos(alpha) + yc * sin(alpha);

% Eliminate negative radii
    if r < 0
        alpha = alpha + pi;
        if alpha > pi
            alpha = alpha - 2 * pi;
        end
        r = -r;
    end
end
```