

Atividade 3 – Caracterização de sensores e “line fitting”

Objetivo

O objetivo desta atividade é realizar a extração dos dados de sensor laser no laboratório e implementar o algoritmo split-and-merge no Matlab para se ajustar os dados de medição à uma linha reta.

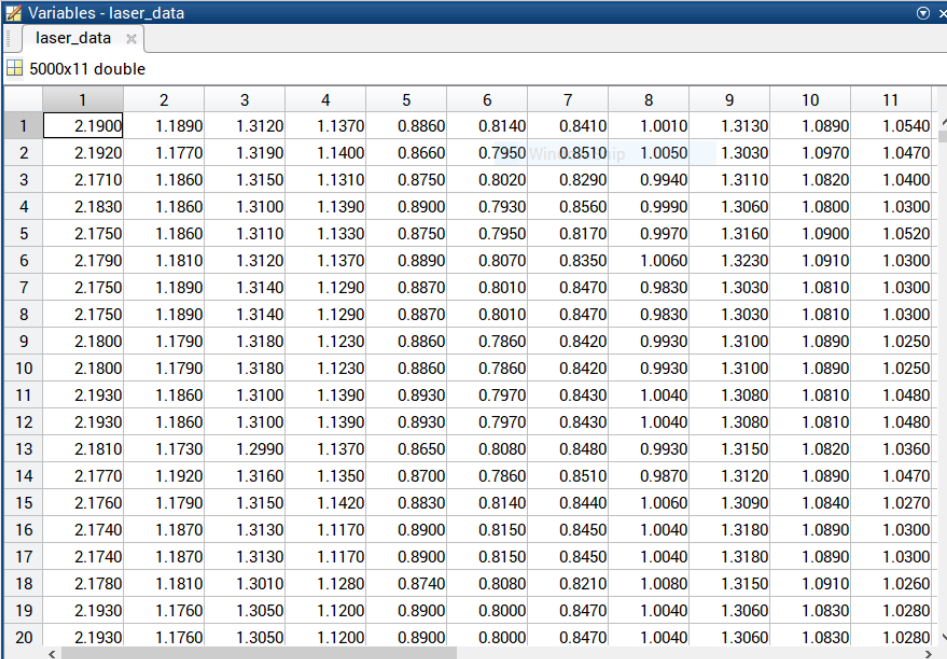
Tarefas 1 e 2

Nestas duas primeiras tarefas foi realizada a extração dos dados do sensor laser do robô no laboratório e caracterização dos dados.

Para tanto, o código Matlab foi integrado com a API Java conforme descrito no tutorial e o código abaixo foi utilizado para executar a leitura dos sensores laser.

```
1. %% reading laser
2. %%PUT YOUR CODE HERE
3. optionStr= '?range=-100:100:20';
4. % Collect data n times and store to laser data array
5. for a = 1:5000
6.     dist = Pioneer_p3dx_getLaserData(connection,'distances');
7.     fprintf("Measurement %d \n", a);
8.     laser_data(a, :) = dist;
9. end
```

O resultado pode ser verificado na matrix laser_data, que consiste de 11 conjuntos de medições do laser com variações de ângulo de 20 graus. Foram realizadas 5000 execuções para se obter uma massa de dados grande.

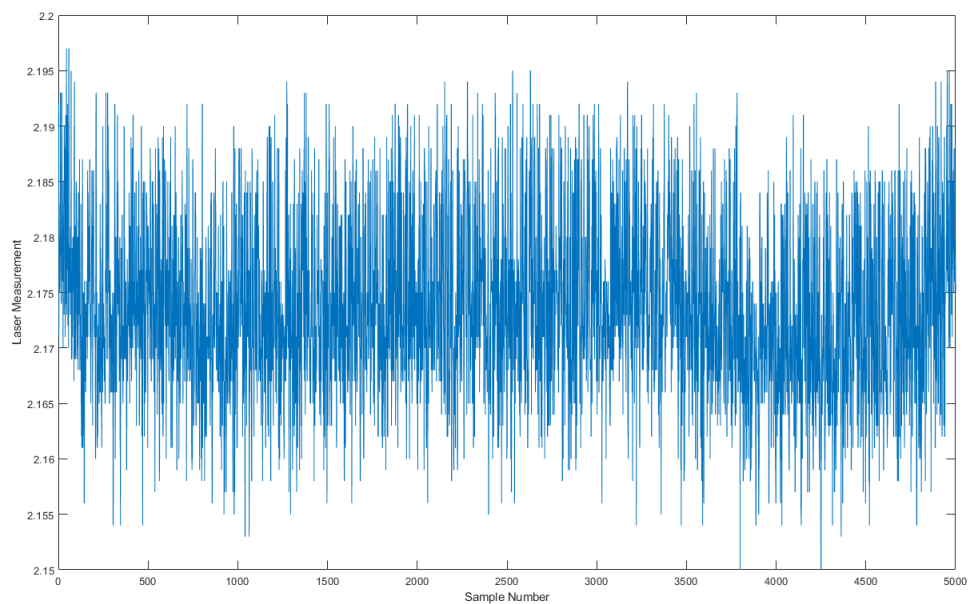


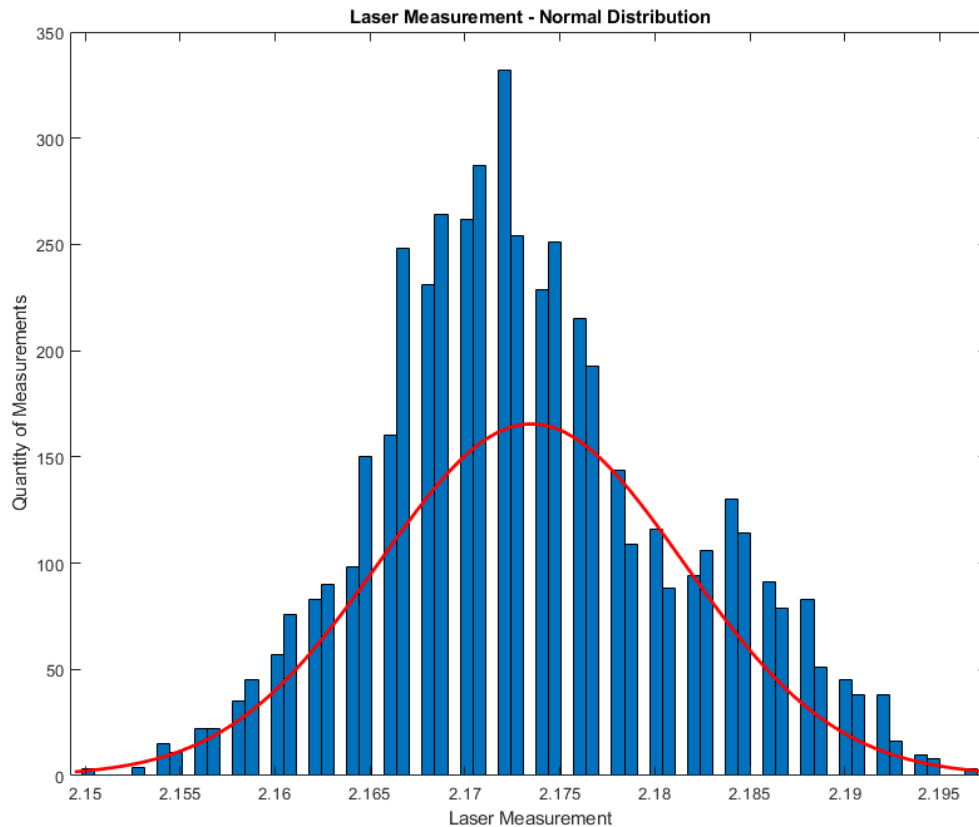
	1	2	3	4	5	6	7	8	9	10	11
1	2.1900	1.1890	1.3120	1.1370	0.8860	0.8140	0.8410	1.0010	1.3130	1.0890	1.0540
2	2.1920	1.1770	1.3190	1.1400	0.8660	0.7950	0.8510	1.0050	1.3030	1.0970	1.0470
3	2.1710	1.1860	1.3150	1.1310	0.8750	0.8020	0.8290	0.9940	1.3110	1.0820	1.0400
4	2.1830	1.1860	1.3100	1.1390	0.8900	0.7930	0.8560	0.9990	1.3060	1.0800	1.0300
5	2.1750	1.1860	1.3110	1.1330	0.8750	0.7950	0.8170	0.9970	1.3160	1.0900	1.0520
6	2.1790	1.1810	1.3120	1.1370	0.8890	0.8070	0.8350	1.0060	1.3230	1.0910	1.0300
7	2.1750	1.1890	1.3140	1.1290	0.8870	0.8010	0.8470	0.9830	1.3030	1.0810	1.0300
8	2.1750	1.1890	1.3140	1.1290	0.8870	0.8010	0.8470	0.9830	1.3030	1.0810	1.0300
9	2.1800	1.1790	1.3180	1.1230	0.8860	0.7860	0.8420	0.9930	1.3100	1.0890	1.0250
10	2.1800	1.1790	1.3180	1.1230	0.8860	0.7860	0.8420	0.9930	1.3100	1.0890	1.0250
11	2.1930	1.1860	1.3100	1.1390	0.8930	0.7970	0.8430	1.0040	1.3080	1.0810	1.0480
12	2.1930	1.1860	1.3100	1.1390	0.8930	0.7970	0.8430	1.0040	1.3080	1.0810	1.0480
13	2.1810	1.1730	1.2990	1.1370	0.8650	0.8080	0.8480	0.9930	1.3150	1.0820	1.0360
14	2.1770	1.1920	1.3160	1.1350	0.8700	0.7860	0.8510	0.9870	1.3120	1.0890	1.0470
15	2.1760	1.1790	1.3150	1.1420	0.8830	0.8140	0.8440	1.0060	1.3090	1.0840	1.0270
16	2.1740	1.1870	1.3130	1.1170	0.8900	0.8150	0.8450	1.0040	1.3180	1.0890	1.0300
17	2.1740	1.1870	1.3130	1.1170	0.8900	0.8150	0.8450	1.0040	1.3180	1.0890	1.0300
18	2.1780	1.1810	1.3010	1.1280	0.8740	0.8080	0.8210	1.0080	1.3150	1.0910	1.0260
19	2.1930	1.1760	1.3050	1.1200	0.8900	0.8000	0.8470	1.0040	1.3060	1.0830	1.0280
20	2.1930	1.1760	1.3050	1.1200	0.8900	0.8000	0.8470	1.0040	1.3060	1.0830	1.0280

As medições obtidas foram verificadas utilizando o teste de Shapiro-Wilk para variações do parâmetro alpha (nível de significância) conforme abaixo:

```
1. >> run_swtest(a)
2. Alpha: 0.050000 Swtest: 1
3. Alpha: 0.100000 Swtest: 1
4. Alpha: 0.150000 Swtest: 1
5. Alpha: 0.200000 Swtest: 1
6. Alpha: 0.250000 Swtest: 1
7. Alpha: 0.300000 Swtest: 1
8. Alpha: 0.350000 Swtest: 1
9. Alpha: 0.400000 Swtest: 1
10. Alpha: 0.450000 Swtest: 1
11. Alpha: 0.500000 Swtest: 1
12. Alpha: 0.550000 Swtest: 1
13. Alpha: 0.600000 Swtest: 1
14. Alpha: 0.650000 Swtest: 1
15. Alpha: 0.700000 Swtest: 1
16. Alpha: 0.750000 Swtest: 1
17. Alpha: 0.800000 Swtest: 1
18. Alpha: 0.850000 Swtest: 1
19. Alpha: 0.900000 Swtest: 1
20. Alpha: 0.950000 Swtest: 1
```

As 5000 medidas foram plotadas nos gráficos abaixo e pode-se verificar que se aproximam de uma distribuição normal.





Tarefa 3

Nesta tarefa o algoritmo de split and merge foi utilizado para extrair linhas das medições de laser utilizando o line fitting das coordenadas de entrada.

Para tanto o código da função `fitLine()` foi alterado conforme descrito no texto:

```

1. function [alpha, r] = fitLine(XY)
2. % Compute the centroid of the point set (xmw, ymw) considering that
3. % the centroid of a finite set of points can be computed as
4. % the arithmetic mean of each coordinate of the points.
5.
6. % XY(1,:) contains x position of the points
7. % XY(2,:) contains y position of the points
8.
9.
10. X = XY(1,:);
11. Y = XY(2,:);
12.
13. xc = mean(X);
14. yc = mean(Y);
15.
16. % compute parameter alpha (see exercise pages)
17. s = 0;
18. for i = 1:length(X)
19.     s = s + (X(i) - xc)*(Y(i) - yc);
20. end
21. nom = -2*s;
22.
23. denom = 0;
24. for i = 1:length(X)
25.     denom = denom + ((Y(i) - yc)^2 - (X(i) - xc)^2);

```

```
26.     end
27.
28.     alpha = atan2(nom,denom)/2;
29.
30.     % compute parameter r (see exercise pages)
31.     % (xc,yc) must lie in the line we are fitting.
32.     r = xc*cos(alpha) + yc*sin(alpha);
```

Nota-se que para obter o parâmetro r foi necessário assumir que o centróide, ponto definido pelas médias das coordenadas X e Y , está situado na reta na qual estamos executando o algoritmo de line fitting.

Os resultados obtidos executando

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

Conclusão

Os exercícios propostos permitiram exercitar a extração de dados do sensor laser em laboratório e implementar o algoritmo de split-and-merge no Matlab com sucesso.