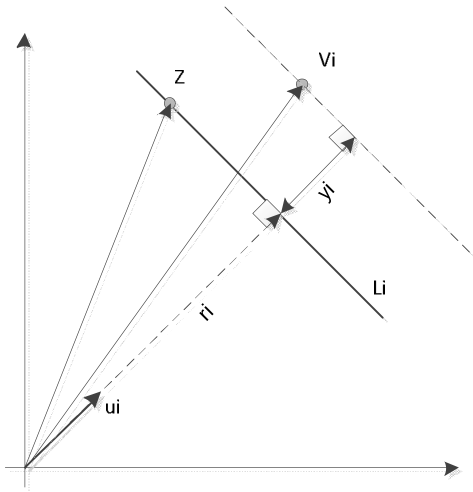
Comments on exercise 3(4)

This is some additional comment on exercise 3 (and 4)

Cox algorithm

This is a short description on the different steps in the Cox scan matching algorithm. For all details refer to the Cox paper!

Step 0 – Calculate all unit vectors of all lines



- Calculate all unit vectors, u_i, off all lines, L_i.
- Calculate distance to line (from origin)

  $r_i = u_i \bullet z$

  there z is any point on the line (select endpoint) and • is the dot-product. In matlab it will look like:

  ```
  ri = dot(ui,Li)
  ```

The loop

Step 1 – translate and rotate data points

- Sensor values to Cartesian coordinates (sensor coordinates)

  ```
  x = r*cos(alfa);
  y = r*sin(alfa);
  ```

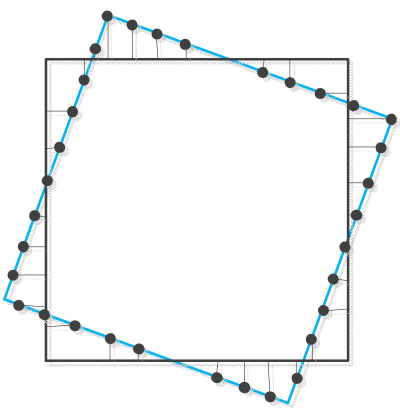
- Sensor coordinates to robot coordinates

  ```
  R = [cos(Sa) –sin(Sa) Sx; sin(Sa) cos(Sa) Sy;0 0 1]
  Xs = R*[x y 1]ᵀ
  ```

- Robot coordinates to world coordinates

  ```
  R = [cos(θ) –sin(θ) Xr(1); sin(θ) cos(θ) Xr(2);0 0 1]
  Xw = R*[Xs(1) Xs(2) 1]ᵀ
  ```

Step 2 – Find the target for data points



Assign points to line, i.e. points to closest lines. This means finding `min(yi)` for all lines L_i and assign point v_i to line L_i. `yi = ri – dot(ui,vi)`

Reject all outliers!

Step 3 – Set up linear equation system

Set up the linear equation system and find `b = (dx, dy, da)` from least square.

```
xi1 = ui1;
xi2 = ui2;
xi3 = ui*[0 -1;1 0]*(vi-vm);
A = [xi1 xi2 xi3];
B = inv(A^TA)A^T*Y
```

Calculate the variance

`S2 =(y-Ab)(y-Ab)/(n-4)` there `n = max(size(A));`

Step 4 - Add latest contribution to the overall congruence

B = [dx dy dθ]

```
ddx = ddx + dx;
ddy = ddy + dy;
ddθ = ddθ + dθ;
```

Step 5 – Check if the process has converged

Stop if the changes is below a threshold.

```
if (sqrt(B(1)^2 + B(2)^2) < 5 )&&(abs(B(3) <0.1*pi/180) ),
% stop loop
else
% return to Step 1
end
```

General comments on Exercise 3 and 4

On exercise 3&4 it is recommended that you plot the error, i.e. true position - estimated position separately for X, Y and Theta using subplot. This code may help you!

```
ERROR = [X' Y' A'] - CONTROL(:,4:6);
ERROR(:,3) = AngDifference(A',CONTROL(:,6));
ERROR = abs(ERROR);
figure,
subplot(3,1,1);
plot(ERROR(:,1),'b'); hold;
plot(sqrt(P(:,1)),'r'); % one sigma
plot(ScanPosIndx,sqrt(P(ScanPosIndx,1)),'k.'); % one sigma
title('Error X [mm] and uncertainty [std] (red)');

subplot(3,1,2);
plot(ERROR(:,2),'b'); hold;
plot(sqrt(P(:,5)),'r'); % one sigma
title('Error Y [mm] and uncertainty [std] (red)');

subplot(3,1,3);
plot(ERROR(:,3)*180/pi,'b'); hold;
plot(sqrt(P(:,9))*180/pi,'r'); % one sigma
title('Error A [degree] and uncertainty [std] (red)');
```