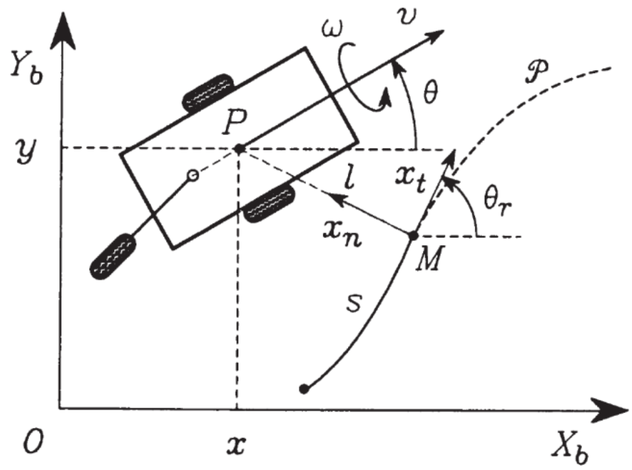
METODY MATEMATYCZNE AUTOMATYKI I ROBOTYKI – ćwiczenie 2

Sterowanie robotem mobilnym – śledzenie ścieżki

Celem ćwiczenia jest symulacyjne zbadanie algorytmu śledzenia ścieżki przez robot mobilny. Zadanie jest zilustrowane na rys. 3.1.



Rys. 3.1. Śledzenie ścieżki przez robot mobilny – model problemu [1]

W ćwiczeniu należy przyjąć, że robot porusza się ze stałą prędkością liniową v i jest sterowany przez zmianę prędkości kątowej ω. Przy takim założeniu można zastosować liniowy lub nieliniowy wariant algorytmu śledzenia zaproponowany w opracowaniu [1]

$$\omega = u + \frac{c}{1-cl} v \cos\tilde{\theta}, \quad u_{lin} = -k_2 vl - k_3 v\tilde{\theta}, \quad u_{nlin} = -k_2 vl \frac{\sin\tilde{\theta}}{\tilde{\theta}} - k_3 v\tilde{\theta}, \tag{1}$$

gdzie: $\tilde{\theta} = \theta - \theta_r$. Wielkości występujące w (1) są oznaczone na rys. 3.1, przy czym M jest punktem śledzonej ścieżki najbliższym względem P, a długość l jest liczona ze znakiem. Wielkość c to krzywizna ścieżki w punkcie M. Parametry algorytmu: $k_2 = a^2$, $k_3 = 2\xi a$, gdzie a decyduje o czasie narastania odpowiedzi dynamicznej oraz ξ jest jej współczynnikiem tłumienia. Kod symulacyjny dla Matlab-a (bez opisu śledzonej ścieżki) został podany poniżej.

```
v=0.1; a=4; ksi=1/sqrt(2);
dt=0.001; plotStep=100;

k2=a^2; k3=2*ksi*a;
pos=initPos; th=initTh;
w=0; posSv=[];

xmin=min(pth(:,1)); xmax=max(pth(:,1));
ymin=min(pth(:,2)); ymax=max(pth(:,2));
marg=0.1; xd=xmax-xmin; yd=ymax-ymin; crad=0.1*min([xd,yd]);
plot(pth(:,1),pth(:,2)); hold on;
circ=rectangle('Position',[pos(1)-crad/2,pos(2)-crad/2,crad,crad],...
    'Curvature',[1,1],'LineWidth',0.5);
axis([xmin-marg*xd,xmax+marg*xd,ymin-marg*yd,ymax+marg*yd]);
axis equal; grid;

cnt=1;
while(norm(pos-pth(end,:))>0.01)
    posSv=[posSv;pos];
    th=th+w*dt; pos=pos+[cos(th),sin(th)]*v*dt;

    [k,l,thr,crv]=track(pos,pth);
    dth=th-thr; dth=mod(dth+pi,2*pi)-pi;
    u=-k2*v*l*sinc(dth/pi)-k3*v*dth;  % u=-k2*v*l-k3*v*dth; (alg. liniowy)
    w=u+v*cos(dth)*crv/(1-l*crv);
```

```
    cnt=cnt+1;
    if cnt>plotStep
        set(circ,'Position',[pos(1)-crad/2,pos(2)-crad/2,crad,crad]);
        plot([pos(1) pth(k,1)],[pos(2) pth(k,2)],'g');
        plot(pth(k,1),pth(k,2),'.b','MarkerSize',10);
        plot(pos(1),pos(2),'.r','MarkerSize',10);
        plot(posSv(:,1),posSv(:,2),'r','LineWidth',1.5);
        posSv=[]; cnt=1; pause(plotStep*dt);
    end
end
hold off;

function [k,l,thr,crv]=track(pos,pth)
    [k,l]=dsearchn(pth,pos);
    if k==1, k=2; elseif k==length(pth), k=k-1; end
    a=pth(k-1,:); b=pth(k,:); c=pth(k+1,:);
    u=[b-a,0]; v=[pos-a,0]; w=[c-b,0];
    cr=cross(u,v); lsgn=sign(cr(3)); cr=cross(u,w); csgn=sign(cr(3));
    thr=angle((b-a)*[1;1i]); l=l*lsgn;
    crv=2*sqrt(1-((a-b)*(c-b)'/(norm(a-b)*norm(c-b)))^2)/norm(c-a);
    crv=csgn*real(crv); crvLim=1000; crv=min(max(crv,-crvLim),crvLim);
end
```

1. Przeanalizuj i objaśnij kod symulacyjny, przede wszystkim: (a) zinterpretuj znaczenie najważniejszych zmiennych: `th`, `thr`, `dth`, `pos`, `pth`, `crv`, `l`, `v`, `w`, (b) omów w jaki sposób wyznaczane są wielkości występujące w formułach (1) i gdzie znajdują się obliczenia wyrażone przez te formuły, (c) omów fragmenty kodu odpowiedzialne za obliczenia kinematyki ruchu robota, (d) wskaż części kodu realizujące animację śledzenia.
2. Na początku algorytmu dodaj kod inicjalizujący zmienne `pth`, `initPos`, `initTh` tak, aby trajektoria śledzona była krzywą parametryczną postaci: $x = 0.9t$, $y = 0.5t\sin(10t)$, $t \in [0,1]$, a stan początkowy robota: $x_0 = 0$, $y_0 = -0.1$, $\theta_0 = -\pi/2$.
3. Przeprowadź symulację śledzenia trajektorii dla wprowadzonych wartości parametrów. Czy robot skutecznie odnajduje ścieżkę? Wyjaśnij znaczenie elementów animacji wyświetlanych kolorem niebieskim, czerwonym oraz zielonym.
4. Zmieniając wartości parametrów v, a i ξ (poprawne są wartości dodatnie), sprawdź, jak każdy z nich wpływa na przebieg śledzenia. Czy można dobrać parametry tak, aby algorytm utracił zbieżność śledzenia dla zadanej trajektorii? – zaprezentuj przykład.
5. Przywróć domyślne wartości v, a oraz ξ i zmodyfikuj program tak, aby stan początkowy robota był inicjalizowany losowo: $x_0 = U(0, 0.3)$, $y_0 = U(-0.1, 0.2)$, $\theta_0 = U(0, 2\pi)$. Przeprowadź serię symulacji. Czy zmiana samych warunków początkowych wpływa na zbieżność śledzenia? – zaprezentuj przykłady.
6. Przeprowadź symulacje śledzenia trajektorii o innych kształtach, np. okrąg, elipsa, spirala Archimedesa, spirala logarytmiczna itp.
7. Rozważ możliwość zastosowania badanego algorytmu do sterowania rzeczywistym robotem. Czy założenia przyjęte w symulacjach są realistyczne? Jakie dodatkowe czynniki należałoby uwzględnić w implementacji rzeczywistej? Czy zakresy możliwych zmian wartości ω, l i $\tilde{\theta}$ będą podlegały ograniczeniom w przypadku konstrukcji rzeczywistej i, jeśli tak, z jakich przyczyn?

[1] C. Canudas de Wit, B. Siciliano, G. Bastin, *Theory of Robot Control*, Springer, London, 1997