Mechatronika segédlet

9. gyakorlat

2017. április 20.

Vadai Gergely, Faragó Dénes

Tartalom

Ha a jegyzetben bármilyen hibát találsz, kérlek jelezd a farago.denes@stud.u-szeged.hu mailcímen.

Feladatleírás

A feladat egy differenciál hajtással rendelkező robot szimulációját lehetővé tevő környezet egy részének megismerése.

plotsegment

A `plotsegment` szakaszok megrajzolásához szükséges pontok előállítására szolgál. Kimenete X és Y koordináták vektorai.

```
function [Xs, Ys] = plotsegment(X1, Y1, X2, Y2, N)

%[0, 1/N, 2/N, ..., 1] vektor:
r = 0:1/N:1;

%X vektor:
Xs=X1+r*(X2-X1);

%Y vektor:
Ys=Y1+r*(Y2-Y1);

% plot(Xs, Ys, '.b');
% axis equal;
end
```

Paraméterei

(`X1, Y1`): az egyenes kezdőpontja

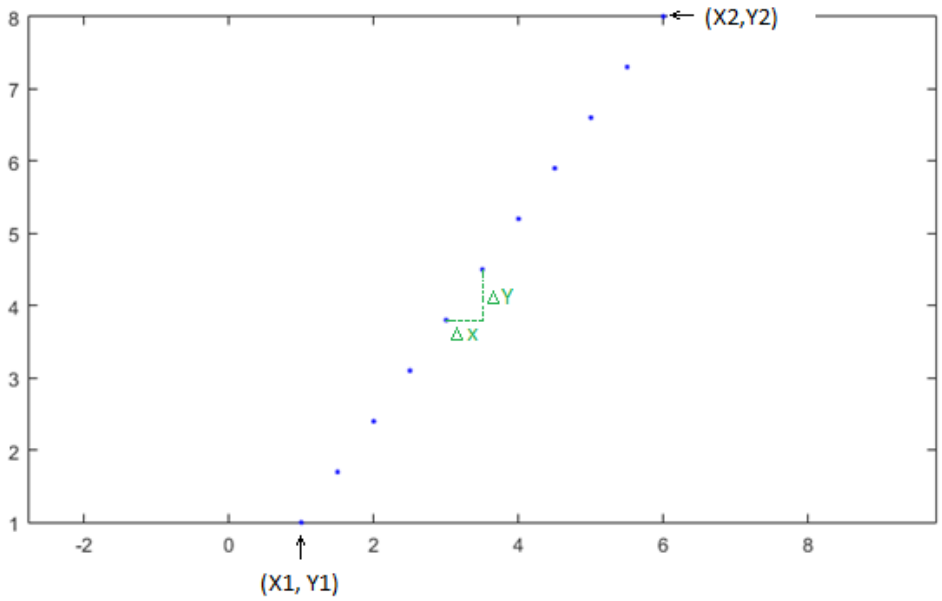
(`X2, Y2`): az egyenes végpontja

`N`: kirajzolt pontok száma – 1

Visszatérési értéke

A szakaszt alkotó `N`+1 darab pont.

A függvény a kezdő- és végpont közötti vízszintes és függőleges távolságot osztja *N+1* részre, majd két hányadost hozzáadja a kezdőpont megfelelő koordinátáihoz *[0, 1, ..., N]*-szer.



1. ábra: plotsegment(1,1,6,8,10) eredménye. (Az egyenes 11 pontból áll.)

drawarc

A `drawarc` ív rajzoláshoz szükséges pontokat állítja elő. Kimenete X és Y koordináták vektora.

```
function [Xs, Ys] = drawarc(X1, Y1, r, AngleDeg, OrientDeg, Components)
v=(OrientDeg-AngleDeg/2):AngleDeg/(Components-1):(OrientDeg+AngleDeg/2);
Xs=X1+r*cosd(v);
Ys=Y1+r*sind(v);
% plot(Xs, Ys, ' .b');
% axis equal;
end
```

Paraméterei

(`X1, Y1`): az ív középpontja

`r`: az ív sugara

`AngleDeg`: az ív nagysága fokban

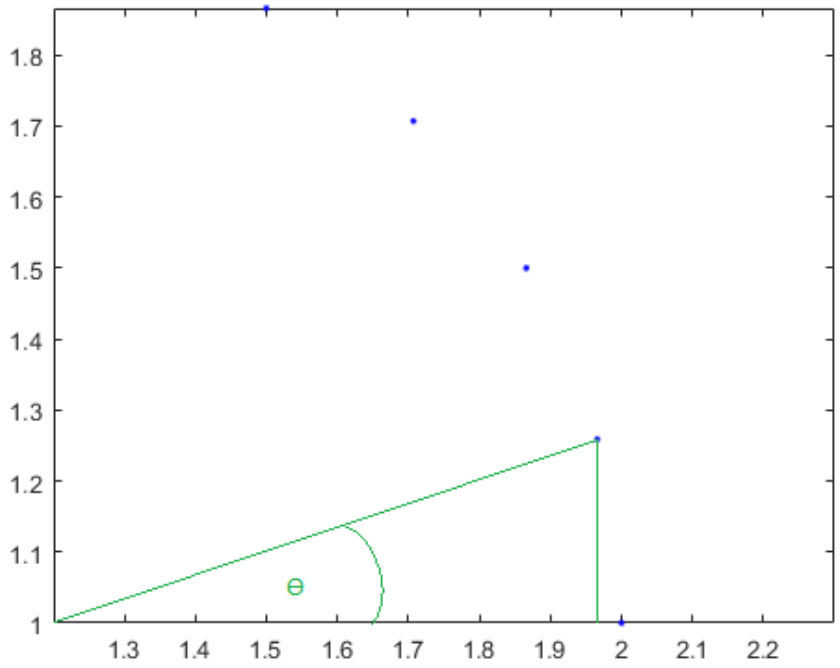
`OrientDeg`: az ív orientációja a vízszintes tengelyhez képest

`Components`: a kirajzolt pontok száma

Visszatérési értéke

A paraméterként kapott ívet alkotó `Components` darab pont.

A működése hasonlít a `plotsegment` függvényhez, csak itt nem a távolságokat osztjuk egyenlő részekre, hanem az ívhez tartozó szöget.



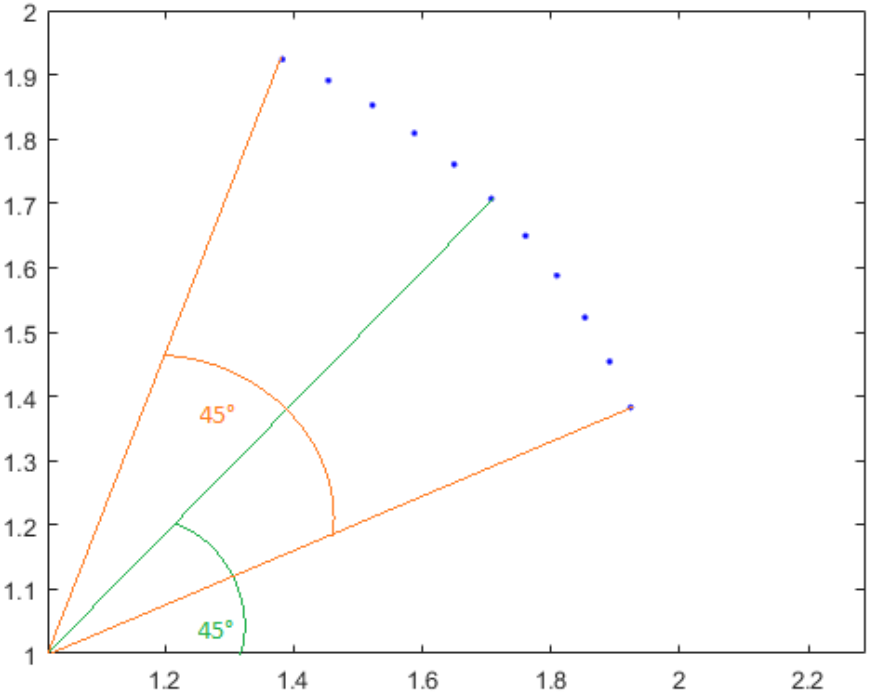
2. ábra: Egy ív pontjai

A 2. ábra pontjainak kiszámítása:

$$x = r \cdot \cos(\theta)$$

$$y = r \cdot \sin(\theta)$$

Θ értéke `AngleDeg/Components` egész számú többszöröse. Az így kapott ívet még el kell forgatni `OrientDeg`gel és `AngleDeg/2`-vel.



3. ábra: drawarc(1,1,1,45,45,11) eredménye

map1.csv

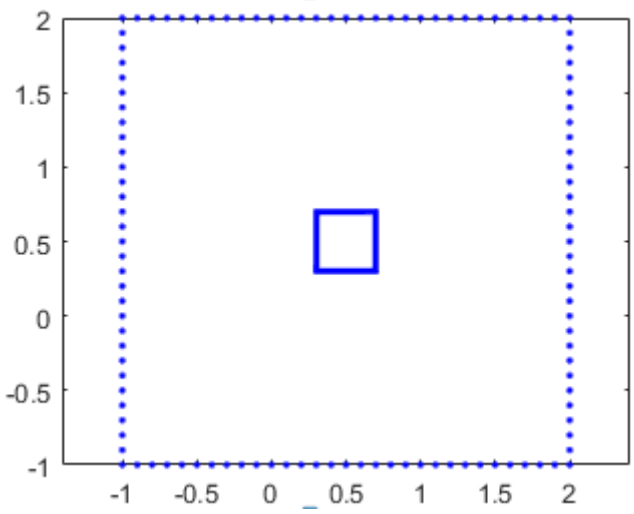
A *map1.csv* a robot által kikerülendő falakat tartalmazza soronként. Minden falat egy (X1, Y1, X2, Y2) szakasz reprezentál.

	X1	Y1	X2	Y2
Keret	-1.0	-1.0	-1.0	2.0
	-1.0	2.0	2.0	2.0
	2.0	2.0	2.0	-1.0
	2.0	-1.0	-1.0	-1.0
Doboz	0.3	0.3	0.3	0.7
	0.3	0.7	0.7	0.7
	0.7	0.7	0.7	0.3
	0.7	0.3	0.3	0.3

1. táblázat: map1.csv sorai

make2dscene

A make2dscene függvény a *csv* fájlban tárolt vonalak pontjait állítja elő a `plotsegment` függvény segítségével. A függvény soronként járja be a *csv* fájlt, és minden sor eredményét hozzáfűzi a kimeneti vektorokhoz.



4. ábra: make2dscene('map1.csv') eredménye

Kód

```
function [Xs, Ys] = make2dscene(filename)
Xs=[];
Ys=[];
M=csvread(filename);
for i=1:size(M, 1),
    [Xn, Yn]=plotsegment(M(i, 1), M(i, 2), M(i, 3), M(i, 4), 30);
    Xs=cat(2, Xs, Xn);
    Ys=cat(2, Ys, Yn);
end
plot(Xs, Ys, ' .b');
```

```
axis equal;
end
```

drawRobot

A `drawRobot` függvény a robot szenzorainak megrajzolására szolgál.

```
function [Xs, Ys] = drawRobot(XP, YP, Orient, SensorOrients, SensorAngle,
SensorRange)

Xs=[];
Ys=[];

for i=1:length(SensorOrients),
    [Xn, Yn]=drawarc(XP, YP, SensorRange, SensorAngle,
Orient+SensorOrients(i), 10);
    Xs=cat(2, Xs, Xn);
    Ys=cat(2, Ys, Yn);
end

plot(Xs, Ys, ' .b');
axis equal;
end
```

Paraméterei

(`XP`, `YP`): a robot középpontja

`Orient`: a robot orientációja

`SensorOrients`: egy vektor, ami a roboton lévő szenzorok orientációit tartalmazza

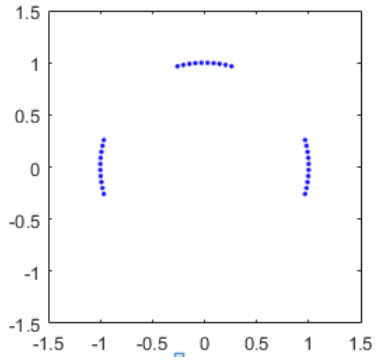
`SensorAngle`: a robot szenzorainak nagysága fokban

`SensorRange`: a szenzorok „látótávolsága”

Visszatérési értéke

A robot szenzorainak kirajzolásához szükséges pontok vektora.

A `SensorOrients` hosszának megfelelő darabszámú ívet rajzol (XP, YP) középponttal, `SensorRange` sugárral, és a robothoz képest `SensorOrients` orientációkkal.



5. ábra: drawRobot(0, 0, 90, [-90, 0, 90], 30, 1) futtatásának eredménye

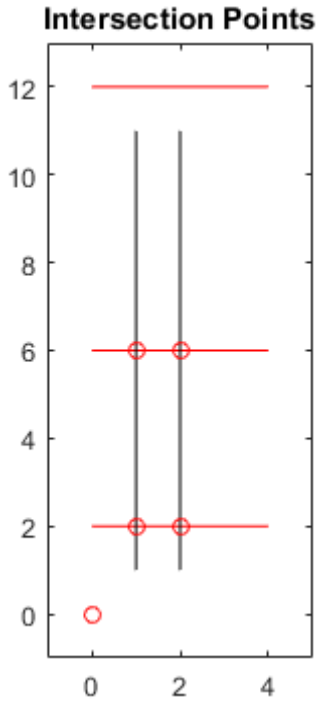
lineSegmentIntersect

A `lineSegmentIntersect` függvény két vonalhalmaz metszéspontjainak kiszámítására szolgál.

```
out = lineSegmentIntersect(XY1,XY2)
```

Paraméterei

Két vonalhalmaz. Egy vonalhalmazt egy N×4-es mátrixként kell megadni, minden sornak egy (X1, Y1, X2, Y2) vonalat kell tartalmaznia. A kimenete egy struktúra, amely megadja, hogy az első vonalhalmazból melyik vonalhalmaz metszi a második vonalhalmaz melyik elemét és hol.



6. ábra: plotsegment

Kimenet:

out.intAdjacencyMatrix

Ha az *i*-edik sorában és a *j*-edik oszlopában 1-es van, az azt jelenti, hogy az első halmaz *i*-edik eleme metszi a második halmaz *j*-edik elemét.

A 7. ábra alapján elmondható, hogy az első halmaz első eleme metszi a második halmaz első és második vonalát, az első halmaz második eleme szintén metszi a második halmaz mindkét szakaszát. Az első halmaz harmadik szakasza nem metsz semmit.

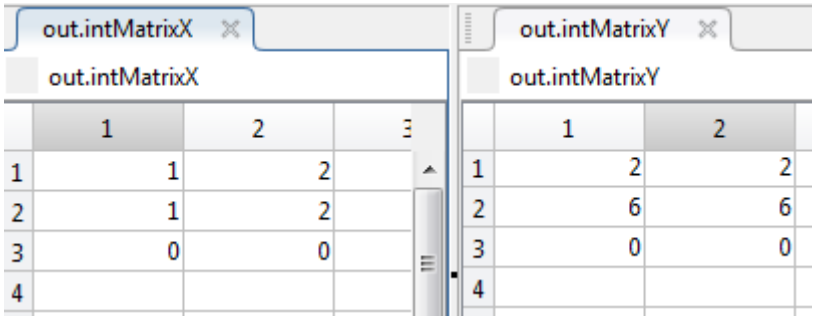
out.intAdjacencyMatrix

1	2
1	1
1	1
0	0

7. ábra: intAdjacencyMatrix

out.intMatrixX és out.intMatrixY

A metszéspontok abszolút koordinátáit tartalmazzák. Ha a szomszédsági mátrixban az *i*-edik sorban és a *j*-edik oszlopban 1-es van, és ugyanezen a helyen az *intMatrixX*-ben található érték *x*, az *intMatrixY*-mátrixban pedig *y*, akkor az azt jelenti, hogy az első halmaz *i*-edik szakasza a második halmaz *j*-edik elemét az *(x, y)* pontban metszi.



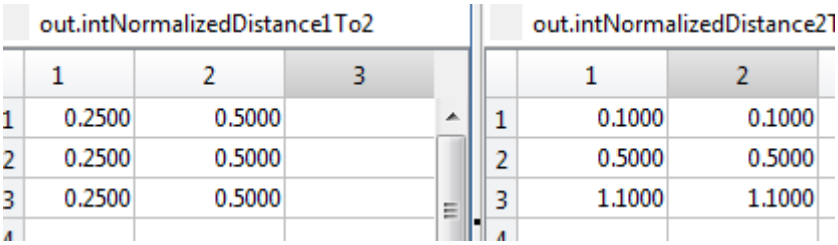
8. ábra: intMatrixX és intMatrixY

out.intNormalizedDistance1To2

Az *i*-edik sorban és a *j*-edik oszlopban lévő szám megadja, hogy az első halmaz *i*-edik szakaszát hol metszi a második halmaz *j*-edik eleme. Ezek az értékek az első halmazban lévő szakaszok hosszaira nézve normalizáltak.

out.intNormalizedDistance2To1

Az *i*-edik sorban és a *j*-edik oszlopban lévő szám megadja, hogy a második halmaz *j*-edik szakaszát hol metszi az első halmaz *i*-edik eleme. Ezek az értékek a második halmazban lévő szakaszok hosszaira nézve normalizáltak.



9. ábra: intNormalizedDistance1To2 és intNormalizedDistance2To1

parAdjacencyMatrix

Ha az *i*-edik sorában és a *j*-edik oszlopában egyes van, akkor az első halmaz *i*-edik szakasza párhuzamos a második halmaz *j*-edik elemével.

coincAdjacencyMatrix

Ha az *i*-edik sorában és a *j*-edik oszlopában egyes van, akkor az első halmaz *i*-edik szakasza egybeesik a második halmaz *j*-edik elemével.

Megjegyzés: az ábrákat az

```
XY1=[0 2 4 2; 0 6 4 6; 0 12 4 12];
XY2=[1 1 1 11; 2 1 2 11];
```

halmazok segítségével készítettem.

Feladatok

Robot középpontjának megrajzolása

Az órán feladat volt a robot középpontjának jelölése egy piros körrel.

Megoldás: `hold on / hold off` parancs segítségével az (`XP, YP`) pont kirajzolása:

```
function [Xs, Ys] = drawRobot(XP, YP, Orient, SensorOrients, SensorAngle,
SensorRange)

Xs=[];
Ys=[];

for i=1:length(SensorOrients),
    [Xn, Yn]=drawarc(XP, YP, SensorRange, SensorAngle,
Orient+SensorOrients(i), 10);
    Xs=cat(2, Xs, Xn);
    Ys=cat(2, Ys, Yn);
end

plot(Xs, Ys, ' .b');
axis equal;
xlim([-1.5, 1.5]);
ylim([-1.5, 1.5]);

hold on;
plot(XP, YP, 'or');
hold off,

end
```

Falak és a robot együttes kirajzolása

Megoldás 1

```
>> make2dscene('map1.csv');
>> hold on;
>> drawRobot(0,0,90,[-90, 0, 90], 30, 0.3);
>> hold off;
```

Megoldás 2

A két függvény pontjainak összefűzésével:

```
>> [Xs1, Ys1] = make2dscene('map1.csv');
>> [Xs2, Ys2] = drawRobot(0,0,90,[-90, 0, 90], 30, 0.3);
>> plot(cat(2, Xs1, Xs2), cat(2, Ys1, Ys2), '.b')
```