Portál pre odborné publikovanie ISSN 1338-0087

20. Matlab – polygóny

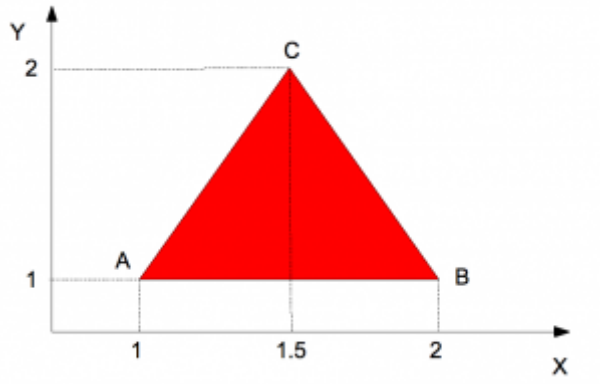
Foltin Martin · MATLAB/Comsol

08.01.2010

V predošlých častiach seriálu boli predstavené postupy, ktoré zobrazovali istú závislosť či už jednej, alebo viac premenných. Časť bola venovaná aj grafom významným pre štatistiku. Dnešná časť ukáže schopnosti Matlabu v oblasti kreslenia uzavretých grafických objektov – *polygónov* a to ako v 2D tak aj v 3D. Dozviete sa ako pomocou atribútov objektov možno vytvárať jednoduché animácie.

Grafická knižnica Matlabu disponuje okrem množstva funkcií na tvorbu grafických závislostí aj príkazom, ktorý vykresluje uzavretý objekt. Je len potrebné zadať množinu súradníc. Polygóny môžu byť využité ako názorné ukážky určitých výsledkov (napr. vyznačenie určitej oblasti). Pre jednoduchosť začneme s 2D polygónom. Najjednoduchším dvojrozmeným grafickým objektom je pravdepodobne trojuholník. Nech sú body, ktoré tvoria jeho vrcholy.

A=[1 ; 1]
B=[2 ; 1]
C=[1,5 ; 2]



Obr. 1. Definovaný trojuholník

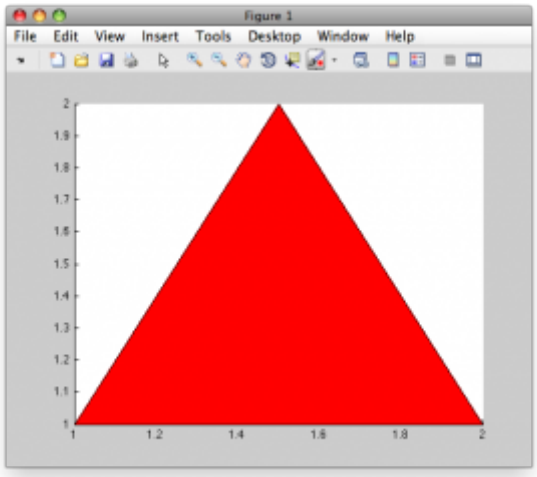
Súradnice bodov môžeme zapísať ako dva vektory.

```
>> x=[1 2 1.5];
>> y=[1 1 2];
```

Keď máme definované súradnice, môžeme pristúpiť ku kresleniu. Polygón sa kreslí príkazom **patch**. Prvé dva parametre sú vektory súradníc bodov. Tretí parameter je farba výplne. Ak teda chceme vykresliť červený trojuholník, zadáme príkaz :

```
>> patch(x,y,'r')
```

Nezabúdajme na tretí parameter, ktorým je farba. V príkaze patch je tento parameter povinný. Výsledok je na obr. 2.



Obr. 2. Trojuholník nakreslený príkazom patch

Nakreslený trojuholník je považovaný za objekt (tak ako aj ostatné grafické objekty v rodičovskom objekte figure). Rovnako ako ostatné objekty, má aj objekt z triedy patch isté vlastnosti, medzi ktoré patria súradnice použitých bodov. Vlastnosť za ktorou sa vrcholy skrývajú sa nazýva *XData* a *YData*. Aktuálne hodnoty môžeme získať pomocou príkazu **get**.

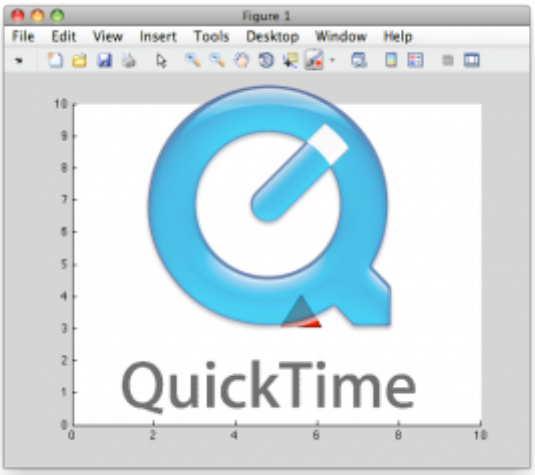
```
>> h=patch(x,y,'r');
>> get(h,'XData')
ans =
1.0000
2.0000
1.5000

>> get(h,'YData')
ans =
1
1
2
```

Keďže dokážeme zisťovať aktuálnu pozíciu bodov, tak sa núka myšlienka aj na zmenu. Na tento účel nám poslúži príkaz **set**. Pomocou tohto príkazu dokážeme zmeniť vlastnosť objektu. Ak sa rozhodneme meniť súradnice bodov, je možné vytvoriť jednoduchú animáciu.

```
figure;
axis([0 10 0 10]);
```

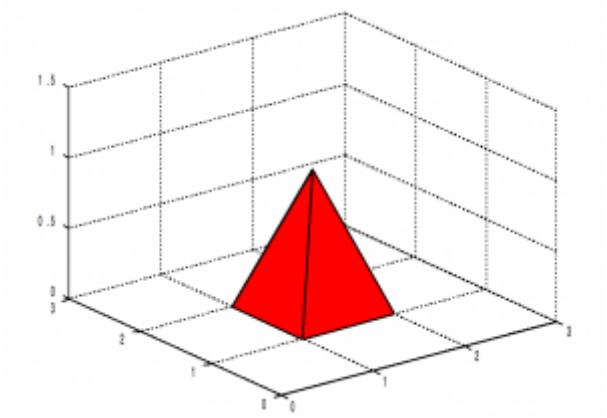
```
x=[1 2 1.5];
y=[1 1 2];
h=patch(x,y,'r');
for xsur=0:0.1:8
x=get(h,'XData');
y=get(h,'YData');
x=x+0.1;
y=y+0.05;
set(h,'XData',x);
set(h,'YData',y);
pause(0.1);
end
```

Rozoberme si ako program funguje. Najskôr príkazom **figure** vytvoríme grafické okno v ktorom sa bude animácia odohrávať. Druhý riadok (príkaz **axis**) definuje rozmer okna. V ďalších 3 riadkoch definujeme súradnice trojuholníka a vytvoríme grafický objekt z triedy patch. Handler na tento objekt bude uložený v premennej *h*. Príkazom **for** otvárame cyklus v ktorom budeme meniť súradnice bodov polygónu. Najskôr získame ich aktuálne pozície (**get**). Následne k x-ovej súradnici pripočítame 0,1 a k y-ovej 0,05. Ďalšie dva riadky zmenia vlastnosť *XData* a *YData* príkazom **set**. Tým pádom sa grafický objekt posunul a malý kúsok vľavo a zároveň hore. Príkaz **pause** slúži na poskytnutie času grafickej vrstve Matlabu, aby stihol prekresliť aktuálnu pozíciu objektu.



Video 1. Pohyb objektu v 2D

Tak ako sme definovali 2D objekt, môžeme definovať aj 3D. Tu je ale situácia mierne zložitejšia. Pri 2D objektoch je jasne definovaná množina bodov ktoré sa pospájajú. Takáto logika sa v 3D nedá uplatniť. V 3D objekte musíme definovať najskôr množinu bodov (tzv. *vertex_list*). Množina bodov sa definuje ako matica s 3 stĺpcami. Každý riadok teda predstavuje jeden bod. Druhá matica definuje spojnice medzi bodmi definovanými vo *vertex_list* (*vertex_connect*). Pokúsme sa vytvoriť grafický objekt podľa obr. 3.



Obr. 3. Štvorboký ihlan

Štvorboký ihlan tvorí 5 bodov. 4 tvoria podstavu (štvorec) a piaty je vrchol. Body teda definujeme takto:

```
>> vertex_list=[2 1 0
2 2 0
1 2 0
1 1 0
1.5 1.5 1];
```

Máme definovaných 5 bodov v 3D. Teraz k nim potrebujeme definovať spojnice, ktoré jednoznačne charakterizujú grafický objekt. Spojníc bude spolu 5. Objekt je tvorený podstavou a 4 bokmi.

```
>> vertex_connect=[1 2 3 4
1 2 5 5
2 3 5 5
4 3 5 5
4 1 5 5];
```

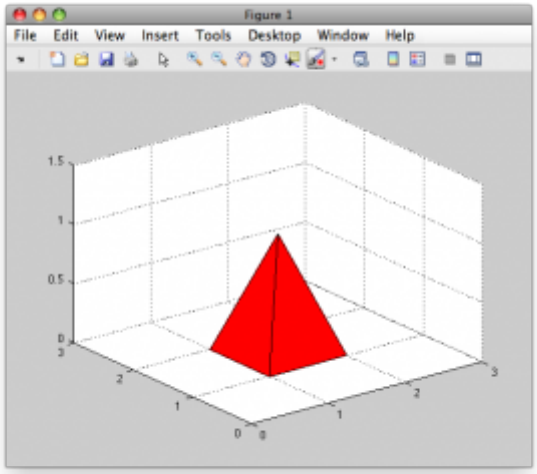
V dvoch maticiach sme si definovali body a ich spojnice. V zmysle obrázku je objekt tvorený čiernymi hranami a červenými plochami. Máme už dostatok informácií a definované potrebné premenné. Môžeme pristúpiť k tvorbe grafického objektu. Príkaz **patch** je teraz zložitejší.

```
>> patch('Vertices',vertex_list,'Faces',vertex_connect,...
'FaceColor','r','EdgeColor','k');
```

Vlastnosť *Vertices* sme naplnili premennou *vertex_list*. Ďalšia vlastnosť požaduje definíciu hrán (*Faces*). Vlastnosť *FaceColor* predstavuje farbu plôch a *EdgeColor* farbu hrán. Aby sme dosiahli rovnaký výsledok ako na obr. 3, musíme definovať rozmer súradnicového systému a pridať mriežku.

```
>> axis([0 3 0 3 0 1.5])
>> grid
```

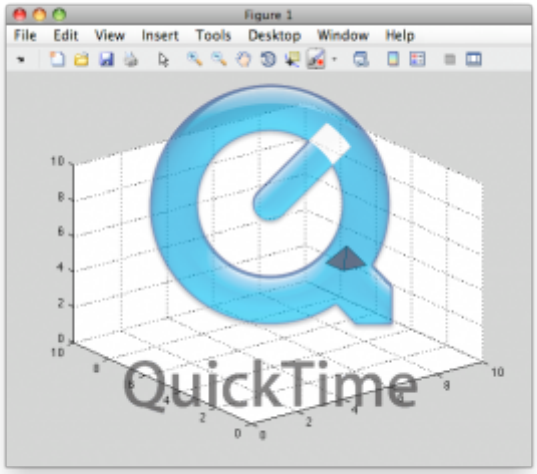
Výsledok je na obr. 4.



Obr. 4. 3D grafický objekt vytvorený príkazom patch

Obdobne ako 2D objekt, je možné animovať aj 3D objekt. Nakoľko spojnice bodov sú definované relatívne, postačuje meniť body definované vo vlastnosti *Vertices*. Vlastnosť *Faces* sa meniť nebude. Princíp animácie je rovnaký ako v prípade 2D. V cykle sa menia súradnice bodov podľa predpísanej funkcie.

```
figure;
axis([0 10 0 10 0 10]);
grid
vertex_list=[2 1 0
2 2 0
1 2 0
1 1 0
1.5 1.5 1]
vertex_connect=[1 2 3 4
1 2 5 5
2 3 5 5
4 3 5 5
4 1 5 5]
h=patch('Vertices',vertex_list,'Faces',vertex_connect,...
'FaceColor','r','EdgeColor','k');
for xsur=0:0.1:8
vertex=get(h,'Vertices');
vertex(:,1)=vertex(:,1)+0.1;
vertex(:,2)=vertex(:,2)+0.07;
vertex(:,3)=vertex(:,3)+0.2*(sin(xsur));
set(h,'Vertices',vertex);
pause(0.1);
end
```



Video 2. Pohyb 3D objektu

Na dvoch príkladoch sme si ukázali možnosti príkazu **patch**. Príkaz môže byť užitočný ak potrebujeme znázorniť jednoduchou animáciou zmeny v procese. Tiež sa dá efektívne využiť pri zvýrazňovaní oblastí v grafe.

Literatúra

1. Matlab 7 – Graphics, The MathWorks, http://www.mathworks.com/access/helpdesk/help/pdf_doc/matlab/graphg.pdf, 24.9.2009