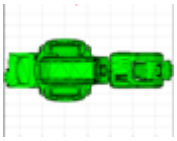
Portál pre odborné publikovanie ISSN 1338-0087

Pokročilá 3D vizualizácia v MATLAB-e

Lešo Martin · MATLAB/Comsol

25.05.2016



Článok sa zaoberá opisom možností 3D vizualizácie objektov tvorených polygónmi v programe MATLAB, ktoré je možné vykresliť v samostatnom grafickom okne. Nakoľko vytvorenie 3D objektov zložitejších tvarov priamo v programe MATLAB môže byť značne komplikovaná úloha, v rámci príspevku je prezentovaná metóda importu zložitejších vizualizácií priamo do MATLAB-u z externých 3D editorov. Článok je zameraný prevažne na použitie funkcie patch pre vykreslenie objektov tvorených pomocou polygónov a na načítanie a spracovanie textových dát v MATLAB-e za použitia regulárnych výrazov. Výsledkom práce je znázornená vytvorená vizualizácia priemyselného robotického ramena ABB IRB – 120.

1. Úvod

V akademickej oblasti, vo výskume a v procese výučby je stály dopyt po tvorbe simulácií existujúcich alebo novo navrhnutých systémov, pričom v súčasnosti prevláda trend prepojenia simulačných modelov systémov s grafickými používateľskými rozhraniami. Neoddeliteľnou súčasťou takýchto aplikácií je zobrazovanie odsimulovaných dát, pričom v prípade tvorby aplikácií v oblasti robotiky, strojárstva a pod. je často požadovaná aj 3D vizualizácia objektov reálneho sveta. V prípade, ak je potrebná podrobná simulácia rôznych modelov systémov a vizualizácie nameraných dát je program MATLAB jednou z najpoužívanejších možností [1-4].

MATLAB je program určený pre numerické výpočty, modelovanie, tvorbu algoritmov, modelovanie, analýzu a vizuálnu prezentáciu dát. Pre vizualizáciu 3D dát ponúka MATLAB rôzne možnosti. Tento príspevok je zameraný na 3D vizualizáciou objektov reálneho sveta. Jednou zo základných funkcií používaných pre 3D vizualizáciu objektov je funkcia patch [5], ktorá umožňuje na základe definovaných bodov (vertices) a polygónov (faces) vytvárať 3D vizualizácie. Táto metóda je vhodná pre vizualizáciu jednoduchých 3D objektov ako sú kocka, ihlan, kužeľ a pod. Vytváranie detailných objektov ručným zápisom bodov a polygónov môže byť často náročná, zdĺhavá až nezvládnuteľná úloha. Z toho dôvodu bolo do programu MATLAB integrovaných mnoho toolboxov pre podporu vytvárania a vizualizácie 3D objektov.

Aplikácia 3D World Editor je prostredie pre vytváranie 3D objektov v MATLAB-e, ktorá je vhodná na vytváranie jednoduchších 3D objektov [6]. MATLAB tiež umožňuje načítavanie 3D objektov z externých programov v spojení so Simulinkom [7,8]. Tieto pokročilejšie riešenia umožňujú pridávanie aj pokročilejších 3D vizualizácií, pričom ich

nevýhodou je ich priama väzba na prostredie Simulink, prípadne jeho knižníc [4]. Používanie takýchto metód vizualizácie môže viesť ku problémom s budúcim prenosom vizualizácií do rôznych 3D aplikácií v prostredí MATLAB z dôvodu nutnosti mať zakúpené a nainštalované dané rozšírenia.

Z vyššie uvedeného rozboru možnosti 3D vizualizácie bolo v rámci riešenia problematiky 3D vizualizácie rôznych komplikovaných objektov odvodené riešenie vizualizácie na základe funkcie patch, pričom jej hlavný nedostatok komplikovaného vytvárania matíc jednotlivých bodov a polygónov bol obídený použitím externého opensource 3D editora Blender [5] pre tvorbu modelov, ktorý v súčasnosti nachádza široké uplatnenie pri tvorbe rôznych vizualizácií, vývoji hier a filmových efektov [9]. Prenos geometrie 3D objektov bol realizovaný exportom objektov vo forme súboru typu OBJ (prípona súboru *.obj) a následným napísaním funkcie na importovanie potrebných dát do programu MATLAB.

2. 3D vizualizácia funkciou patch

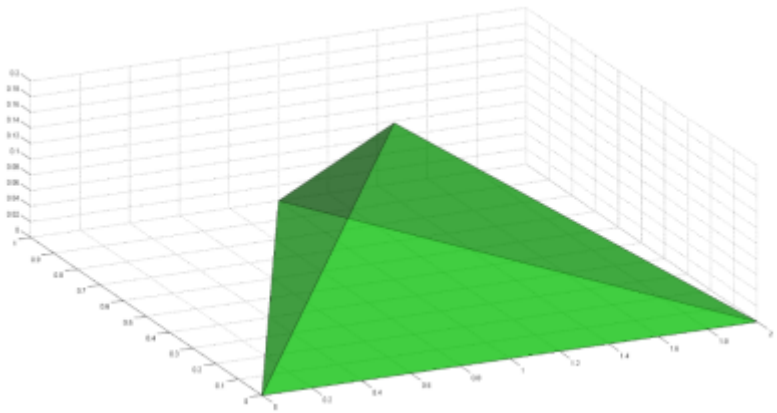
Funkcia patch slúži na vytváranie vizualizácií ľubovoľných objektov, ktoré sú reprezentované ako zoskupenie polygónov, pričom každý z polygónov je definovaný pomocou bodov (vertices) v trojrozmernom priestore. Funkcia patch umožňuje tvorbu zoskupenia polygónov dvoma odlišnými spôsobmi definície jednotlivých bodov a polygónov [7]. Prvým spôsobom opisu 3D objektov je vymenovanie bodov, ktoré tvoria jednotlivé polygóny nasledovným spôsobom:

patch(xdata, ydata, zdata, ‘farba’);

Uvažujme príklad vytvorenia objektu znázorneného na obr. 1:

```
xdata = [[0; 1; 2] [0; 1; 1] [0; 1; 2] [1 ; 1; 2]];
ydata = [[0; 1; 0] [0; 1; .5] [0; .5; 0] [.5; 1; 0]];
zdata = [[0; 0; 0] [0; 0; .2] [0; .2; 0] [.2; 0; 0]];
grid on
```

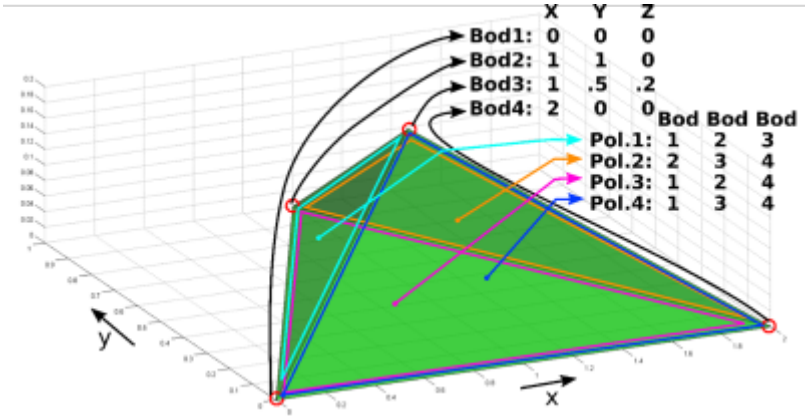
```
h = patch(xdata, ydata, zdata, 'g')
set(h, 'Facealpha', 0.5)
axis equal
% Pridanie osvetlenia
camlight
```



Obr.1 Vizualizácia objektu funkciou patch

Premenná h predstavuje odkaz na vytvorený objekt, pomocou ktorého je možné editovať ľubovoľné vlastnosti (parametre) objektu [1], pričom parameter Facealpha predstavuje mieru priesvitnosti jednotlivých polygónov. Použitím príkazu axis equal dosiahneme zobrazenie vytvoreného objektu v rovnomerne rozloženej mierke pre jednotlivé osi.

Hlavnou nevýhodou takejto definície vizualizácie objektov je, že body tvoriace jednotlivé polygóny sú tvorené duplicitne, čo môže prinášať komplikácie pri vytváraní rozsiahlejších objektov a zároveň výrazne predlžuje dobu prepočtu pri animácii objektov. Ako časté zjednodušenie riešenia tohto problému je separátna definícia bodov v priestore a následná definícia polygónov na základe indexovania jednotlivých bodov v priestore, čím je dosiahnuté, že každý bod v priestore je definovaný iba jedenkrát. Definovanie objektov takýmto spôsobom prináša rýchlejšie spracovanie 3D objektov pri animáciách, ako aj pamäťové úspory. Princíp vytvorenia objektu pomocou definície jednotlivých bodov (vertices) a polygónov (faces) je znázornený na obr. 2:



Obr.2 Princíp vizualizácie funkciou patch so separátnou definíciou bodov a polygónov

V prípade, ak zapíšeme hodnoty z obr. 2 do maticového tvaru, môžeme vykresliť totožnú vizualizáciu ako je zobrazená na obr.1 nasledovne:

```
vertices = [0 0 0; 1 1 0; 1 0.5 0.2; 2 0 0];
faces = [1 2 3; 2 3 4; 1 2 4; 1 3 4];
```

patch('vertices', vertices, 'faces', faces, ...
'Facecolor', [0 1 0], 'Facealpha', 0.5)
grid on
set(h, 'Facealpha', 0.5)
axis equal
% Pridanie osvetlenia
camlight

3. Vizualizácia detailných objektov

Ako bolo spomínané v úvode, vytváranie detailných objektov priamym zapisovaním ich súradníc do matíc, ktoré by pozostávali v rádovo stovkách až tisícich polygónov by bolo časovo náročné a vysoko neefektívne. Z toho dôvodu je jednou z najefektívnejších metód pre vytváranie 3D objektov použitie externých 3D editorov. V súčasnosti existuje mnoho softvérov pre vytváranie 3D objektov, pričom väčšinu z nich môžeme

rozdeliť do dvoch skupín:

Technické 3D CAD programy, ktoré prevažne slúžia na návrh reálnych zariadení a ich vizualizácií ako Solidworks, PTC Creo a pod.- ukladajú vytvorené objekty vo formátoch, ktoré nie sú priamo tvorené pomocou polygónov, pričom väčšinou umožňujú export do niektorých z polygónových formátov .

Do druhej skupiny 3D editorov môžeme zahrnúť 3D vizualizačné programy, ktoré sú prevažne určené na vytváranie objektov virtuálneho sveta, ktoré sú ďalej používané v rámci vývoja hier, filmov a rôznych vizualizácií. K základným vlastnostiam týchto editorov patrí, že medzi základnými modelovacími ponukami existuje priama možnosť vytvárania 3D objektov pomocou polygónov. V súčasnosti existuje niekoľko rôznych editorov, pričom medzi najznámejšie a najpoužívanejšie editory môžeme zahrnúť 3D studio Max, 3D cinema, Autodesk Maya, pričom ako alternatíva týchto komerčných programov je často používaný opensource editor Blender, ktorý predstavuje profesionálny voľne šíriteľný nástroj na tvorbu vizualizácií [8, 9].

Každý z vyššie spomenutých programov ukladá vytvorené vizualizácie vo vlastnom špecifickom formáte, pričom na prenos 3D objektov medzi jednotlivými programami je štandardne zaužívaný wavefront OBJ typ súboru z dôvodu jednoduchej a stručnej definície súboru a spôsobu zápisu dát v ASCII (textovom) formáte. Ako nevýhodu použitia takéhoto typu súboru je paradoxne možno uviesť jeho jednoduchosť a stručnosť, čím je znemožnené prenášať vyššiu funkcionalitu spomínaných editorov [12].

3.1 Importovanie OBJ typu súborov do programu MATLAB

Jednoduchosť a rozšírenosť OBJ súboru [12] viedla k použitiu práve tohto typu súboru pre tvorbu funkcie pre import 3D objektov do programu MATLAB. Skôr ako je možné uvažovať nad písaním samotnej funkcie pre import do MATLAB-u je nutné poznať štruktúru súboru.

Na začiatku súboru je štandardne uvedená hlavička súboru, ktorá je uvedená v rámci komentárov označených znakom #. Za hlavičkou sú uložené dáta o 3D objektoch, pričom typ dát je reprezentovaný kľúčovým slovom, za ktorým sú uvedené jeho parametre, pričom jednotlivé parametre sú oddelené prázdnou medzerou. Zoznam kľúčových slov je uvedený nižšie, pričom pre import geometrie 3D objektov do MATLAB-u sú predovšetkým významné kľúčové slová o, v a f, ktoré reprezentujú názov objektu, pozíciu bodov v priestore a body tvoriace jednotlivé polygóny. Definícia polygónov na základe bodov je indexovaná na základe absolútneho poradia bodov (dôležité pri načítavaní viacerých objektov z jedného súboru).

1. o – názov objektu
2. v – definovanie bodov objektu v priestore
3. vt – súradnice textúr
4. vn – opis normál objektu
5. f – body tvoriace polygóny
6. l – čiara
7. usemtl – meno materiálu

8. mtllib – názov knižnice s materiálmi

Príklad zápisu objektu kocka v OBJ type súboru je nasledovný:

```
# Moja kocka
o Kocka
v 0 0 0
v 1 0 0
v 1 1 0
v 0 1 0
v 0 0 1
v 1 0 1
v 1 1 1
v 0 1 1
f 1 2 6 5
f 2 3 7 6
f 3 4 8 7
f 4 1 5 8
f 1 2 3 4
f 5 6 7 8
```

Tento zápis predstavuje základný opis geometrie objektov v rámci daného štandardu. Mnohé súčasné externé editory exportujú rozšírený zápis o objektoch, v ktorých sú zahrnuté materiály, súradnice normálových vektorov a textúr, ktoré sú označené kľúčovými slovami vn a vt. V takom prípade sa zloženie jednotlivých polygónov zapisuje nasledovným spôsobom:

f index_v/index_vt/index_vn index_v/index_vt/index_vn index_v/index_vt/index_vn

f 1/2/1 2/2/1 3/1/1

pričom znak / slúži ako oddeľovač bodov od bodov textúr a normál.

Problematiku importu takýchto dát do MATLAB-u je možné rozdeliť do troch krokov:

- načítanie textových údajov do programu MATLAB
- spracovanie načítaných údajov
- zápis spracovaných údajov do vhodných dátových typov

Ako prvý krok pri tvorbe funkcie na importovanie 3D objektov vo formáte OBJ typu súboru je načítanie požadovaných textových údajov do MATLAB-u. Pre riešenie tejto úlohy je možné postupovať dvoma rôznymi spôsobmi. Prvou možnosťou je použitie funkcií MATLAB-u pre import ASCII dát a ich zápis vo forme matíc, prípadne štruktúr [14]. Nevýhodou takéhoto riešenia je, že načítané dáta sú vo forme štruktúry, ktorá obsahuje separátne oddelené textové a numerické dáta, čím vzniká komplikácia s budúcim spracovaním údajov.

Ako alternatíva k predošlému postupu je načítanie údajov pomocou funkcií pre načítavanie obsahu súborov ako fopen, fclose, fread a pod. Tieto funkcie sú štandardne používané pre komunikáciu s ľubovoľnými typmi súborov v rôznych programovacích

jazykoch a sú implementované aj v rámci programu MATLAB, pričom predstavujú štandardnú cestu pre načítanie, prípadne zápis dát do súborov. Pre načítanie dát zo súboru je použitá trojica funkcií pre prácu so súbormi a to:

- fopen – otvorenie súboru,
- fread – čítanie údajov zo súboru,
- fclose – uzatvorenie súboru.

Pomocou týchto troch funkcií je možné načítať dáta v textovej podobe do programu MATLAB nasledovne:

```
a=fopen('priklad.obj');
str=rot90(fread(a));
str=char(str);
fclose(a);
```

Druhá a komplikovanejšia časť funkcie je spracovanie týchto údajov, pričom spracovanie jednotlivých údajov sa vykonáva v cykloch. Všeobecne je možné na spracovanie údajov použiť súbor podmienok a funkcií pre prácu s textovými reťazcami, pričom k základným a najčastejšie používaným funkciám v rámci programu MATLAB môžeme priradiť nasledovné funkcie:

- str2num – slúži na konverziu textové reťazca na numerickú hodnotu,
- num2str – slúži na konverziu numerickej hodnoty na textový reťazec,
- strcmp – porovnanie dvoch textových reťazcov.

V prípade snahy o napísanie čo najvšeobecnejšieho algoritmu pre spracovanie dát je preferovanou možnosťou použitie regulárnych výrazov (regular expression), ktoré slúžia na vyhľadávanie určitých vzorov v texte a sú štandardnou súčasťou väčšiny programovacích jazykov [15]. V rámci MATLAB-u je možné využívať funkciu regexp, ktorá slúži na nájdenie daného regulárneho výrazu v texte a následne vykoná spracovanie reťazca na základe zvolenej voľby [16]. V rámci tvorby funkcie pre import objektov bola využívaná voľba split, ktorá reprezentuje požiadavku na rozčlenenie textu pri nájdení zhody so zadaným regulárnym výrazom. V prípade, ak ako regulárny výraz určíme '\n+', daný text bude rozčlenený po jednotlivých riadkoch, pričom budú vynechané prázdne riadky. Takéto rozčlenenie môžeme vykonať nasledovne:

```
file_lines = regexp(str, '\n+', 'split');
```

Takto rozčlenené dáta je následne možné spracovať v cykle, pričom v rámci cyklu je vytváraná štruktúra, do ktorej sa zapisujú súradnice jednotlivých bodov a body tvoriace jednotlivé polygóny. V rámci tohto algoritmu je uvažované, že celý objekt je tvorený polygónmi s rovnakým počtom hrán, pričom vo väčšine prípadov sú polygóny reprezentované ako množina trojuholníkov z dôvodu, že ľubovoľný polygón je možné rozčleniť na trojuholníky. Použitý kód pre spracovanie textových dát o objektoch do štruktúry vyzerá nasledovne:

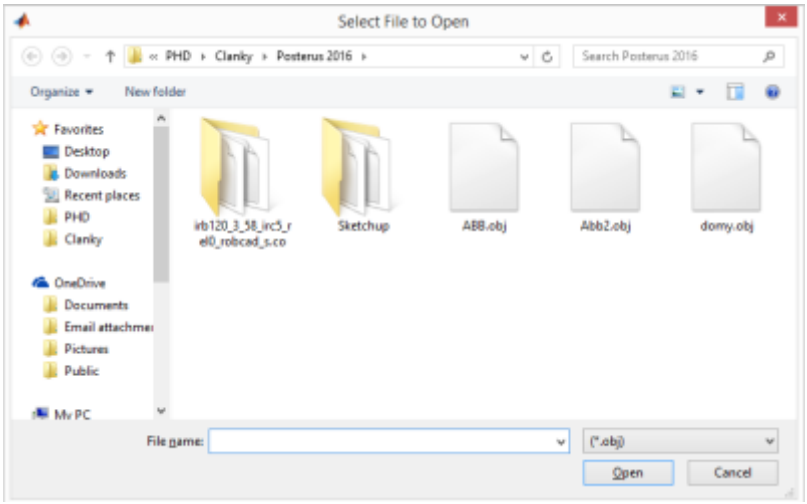
```
for i=1:numel(file_lines)
 file_word = regexp(file_lines{i}, '\s', 'split');
 switch file_word{1}
```

```
  % Novy nazov objektu
  case 'o'
   % Nazov objektu
   cur_obj_name = file_word{2};
   % Inicializacia
   v_abs_last = v_abs;
   v = 0;
   f = 0;
  % Bod
  case 'v'
   % Relativny pocet nacitanych bodov
   v = v +1;
   % Absolutny pocet nacitanych bodov
   v_abs = v_abs + 1;
   objects.(cur_obj_name).vertices(v, 1:3) = ...
   [ str2num(file_word{2}) str2num(file_word{3}) ...
   str2num(file_word{4}) ];
  % Polygon
  case 'f'
   % Relativny pocet nacitanych polygonov
   f = f + 1;
   % pocet bodov polygona
   c = numel(file_word) - 1;
   objects.(cur_obj_name).faces(f, 1:c)= zeros(1, c);
   % zapis bodov
   for j = 1:c
    sub_word = regexp(file_word{j+1}, '/+', 'split')
    objects.(cur_obj_name).faces(f, j) = ...
    str2num(sub_word{1})- v_abs_last;
   end
  otherwise
 end
end
```

pričom algoritmus môže načítať ľubovoľné množstvo objektov, s ľubovoľným počtom bodov tvoriacich polygón pre každý objekt. Výslednú funkciu môžeme zadefinovať v nasledujúcom tvare:

```
function objects = read_obj()
% cesta k súboru
path = uigetfile('*.obj');
```

pričom funkcia neobsahuje žiaden vstupný argument, cesta a názov k súboru sa získava pomocou funkcie uigetfile, ktorá použitím argumentu formátu *.obj zobrazuje všetky viditeľné súbory typu OBJ v aktuálnom priečinku (obr. 3).



Obr. 3 Voľba súboru pre importovanie

3.2 Vykreslenie importovaných súborov

Po importe do programu MATLAB je možné zobraziť importované objekty nami vytvorenou funkciou draw_objects, ktorá vracia odkaz (handle) na vytvorené objekty a grafické okno, v ktorom sú dané objekty vykreslené. Funkcia má jediný povinný vstupný parameter objects, ktorý má obsahovať štruktúru vytvorenú funkciou read_obj. Druhý nepovinný argument handle slúži na definíciu grafického okna, do ktorého chceme dané zobrazenie vykresliť.

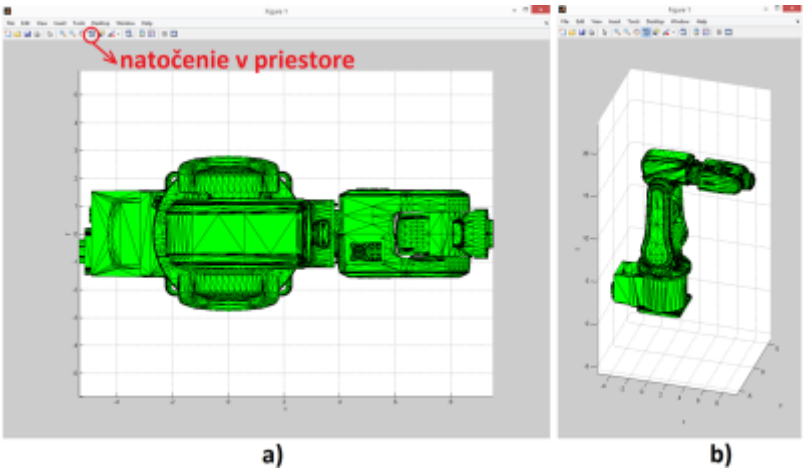
```
function [handles] = draw_objects(objects, handle)
%% vykreli objekty importovane funkciou read_obj
if nargin == 2
 handles.axes = handle;
else
 axes;
 handles.axes = gca;
end

axes(handles.axes);
hold on
name = fieldnames(objects);
name = sort(name);

for i = 1:numel(name)
 % slovne odkazy na objekty
 handles.obj_n.(name{i}) = patch(objects.(name{i}), ...
 'Facecolor', 'g');
 % ciselne odkazy na objekty
 handles.obj_id(i) = handles.obj_n.(name{i});
end
% Nastavenie grafickeho okna
grid on
axis equal
xlabel('x')
zlabel('z')
```

ylabel('y')

Použitím funkcie pre importovanie OBJ súborov je možné načítať ľubovoľný OBJ typ súboru, ktorý je následne možné zobraziť v MATLAB-e pomocou funkcie draw_objects. Na obr. 4 je zobrazená vizualizácia robotického ramena ABB IRB 120 v programeMATLAB na základe vyššie uvedených funkcií.



Obr. 4 Vizualizácia robotického ramena a) pohľad zhora b) pohľad z boku

4. Záver

V rámci článku bola prezentovaná metóda pre zobrazovanie ľubovoľných 3D objektov v rámci programu MATLAB prostredníctvom funkcie patch. Problematika tvorby zložitejších a detailnejších objektov bola vyriešená vytvorením funkcie pre import dát zo súboru typu OBJ, ktorá umožňuje prenos informácií o geometrii ľubovoľných objektov do prostredia MATLAB-u. Takéto riešenie umožňuje jednoduché a rýchle vytvorenie 3D vizualizácií objektov reálneho sveta v rámci programu MATLAB, ktoré je možné integrovať do rôznych grafických používateľských rozhraní.

Zoznam použitej literatúry

1. Fedák V., Záskalický P.: Support for Learning of Dynamic Performance of Electrical Rotating Machines by Virtual Models, E-Learning – Instructional Design, Organizational Strategy and Management, Rijeka : InTech, s. 3-31, 2015.
2. Žilková J., Lešo M., Vacek M.: Riadenie robotov s pohonmi Dynamixel s využitím GUI, TU – 2015, 68 s, ISBN 978-80-553-2435-7.
3. Sivý R., Lešo M., Pajkoš M.: Ovládanie servopohonov Dynamixel pomocou grafického rozhrania, Electrical Engineering and Informatics 6: Proceedings of the Faculty of Electrical Engineering and Informatics of the Technical University of Košice, s. 223-225, 2015.
4. Klasna J.: MATLAB graphical user interface development for education support, Proceedings of the 2011 34th International Spring Seminar on Electronics Technology (ISSE), Tatranská Lomnica, 2011, pp. 645-648.
5. Foltin M.: Matlab – polygóny, Posterus, 3(1), 2010, Online: http://www.posterus.sk/?p=5082
6. Mathworks documentation. Online: http://www.mathworks.com/help/sl3d/the-3d-world-editor.html
7. Mathworks documentation. Online:

http://www.mathworks.com/products/3D-animation/index.html
8. Blender features. Online: https://www.blender.org/features/
9. Veldhuizen B:. Blender Used in Previz for Captain America: the Winter Soldier, Blendernation, 2014, Online: http://www.blendernation.com/2014/05/02/blender-used-in-previz-for-captain-america-the-winter-soldier/
10. Mathworks documentation. Online: http://www.mathworks.com/help/matlab/ref/patch.html
11. Mathworks documentation. Online: http://www.mathworks.com/help/matlab/ref/patch-properties.html
12. Špecifikácia OBJ súboru: http://www.martinreddy.net/gfx/3d/OBJ.spec
13. Mathworks documentation. Online: http://www.mathworks.com/help/matlab/ref/importdata.html
14. Mathworks documentation. Online: http://www.regular-expressions.info/quickstart.html
15. Mathworks documentation. Online: http://www.mathworks.com/help/matlab/ref/regexp.html
16. Lešo, M.: Vizualizácia v robotike, Bakalárska práca, FEI TU v Košiciach, 2012.

Spoluautorom článku je Jaroslava Žilková, Katedra elektrotechniky a mechatroniky, Fakultra elektrotechniky a informatiky, Technická univerzita v Košiciach, Slovenská republika