Automatsko planiranje robotskog kretanja pomoću CAD modela

Košak, Matija

Undergraduate thesis / Završni rad

2018

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **University of Zagreb, Faculty of Mechanical Engineering and Naval Architecture / Sveučilište u Zagrebu, Fakultet strojarstva i brodogradnje**

Permanent link / Trajna poveznica: https://urn.nsk.hr/urn:nbn:hr:235:773748

Rights / Prava: In copyright / Zaštićeno autorskim pravom.

Download date / Datum preuzimanja: **2024-03-28**

Repository / Repozitorij:

Repository of Faculty of Mechanical Engineering and Naval Architecture University of Zagreb

SVEUČILIŠTE U ZAGREBU
FAKULTET STROJARSTVA I BRODOGRADNJE

ZAVRŠNI RAD

Matija Košak

Zagreb, 2018. godina.

SVEUČILIŠTE U ZAGREBU
FAKULTET STROJARSTVA I BRODOGRADNJE

ZAVRŠNI RAD

Mentori:

Prof. dr. sc. Bojan Jerbić, dipl. ing.
Mag. Ing. Mech. Josip Vidaković

Student:

Matija Košak

Zagreb, 2018. godina.

Izjavljujem da sam ovaj rad izradio samostalno koristeći znanja stečena tijekom studija i navedenu literaturu.

Zahvaljujem se svom mentoru Prof. Dr. Sc. Bojanu Jerbiću na pruženoj stručnoj pomoći i savjetima prilikom izrade završnog rada.

Također se želim zahvaliti asistentu Mag. Ing. Mech. Josipu Vidakoviću na korisnim savjetima i pomoći pri izradi završnog rada.

Posebno hvala mojoj obitelji, roditeljima Suzani i Robertu i bratu Davoru te svim bližnjima i kolegama na podršci i pomoći tijekom studija.

Matija Košak

SVEUČILIŠTE U ZAGREBU
FAKULTET STROJARSTVA I BRODOGRADNJE
Središnje povjerenstvo za završne i diplomske ispite
Povjerenstvo za završne ispite studija strojarstva za smjerove:
proizvodno inženjerstvo, računalno inženjerstvo, industrijsko inženjerstvo i menadžment, inženjerstvo materijala i mehatronika i robotika

Sveučilište u Zagrebu Fakultet strojarstva i brodogradnje	
Datum	Prilog
Klasa:	
Ur.broj:	

ZAVRŠNI ZADATAK

Student: **Matija Košak** Mat. br.: 0035199458

Naslov rada na hrvatskom jeziku: **AUTOMATSKO PLANIRANJE ROBOTSKOG KRETANJA POMOĆU CAD MODELA**

Naslov rada na engleskom jeziku: **AUTOMATED ROBOT PATH PLANNING USING CAD MODEL**

Opis zadatka:

U robotskim primjenama, kao što je robotsko poliranje ili brušenje, koje zahtijevaju praćenje površinske geometrije objekata, putanje se mogu generirati pomoću CAD modela. U sklopu zadatka potrebno je razviti algoritam za automatsko generiranje robotskih putanja na osnovu geometrijskih podataka definiranih u CAD softveru (CATIA). Generirane putanje moraju biti prilagođene robotskim upravljačkim zahtjevima. Za razvijeno rješenje izvršiti validaciju putem računalne simulacije te potom u Laboratoriju za projektiranje izradbenih i montažnih sustava na jednom od robota.

Zadatak zadan:
30. studenog 2017.

Rok predaje rada:
1. rok: 23. veljače 2018.
2. rok (izvanredni): 28. lipnja 2018.
3. rok: 21. rujna 2018.

Predviđeni datumi obrane:
1. rok: 26.2. - 2.3. 2018.
2. rok (izvanredni): 2.7. 2018.
3. rok: 24.9. - 28.9. 2018.

Zadatak zadao:

Prof. dr. sc. Bojan Jerbić

Predsjednik Povjerenstva:

Izv. prof. dr. sc. Branko Bauer

SADRŽAJ

SADRŽAJ I

POPIS SLIKA III

POPIS KRATICA V

SAŽETAK VI

SUMMARY VII

1. UVOD 1
   1.1. Pregled područja primjene 1
      1.1.1. Robotsko poliranje 1
      1.1.2. Robotsko brušenje 2
      1.1.3. Robotsko bojanje i lakiranje 4
2. ANALIZA CAD MODELA 5
   2.1. CAD model 5
   2.2. Površina (Surface) 5
   2.3. Stvaranje točaka i "meshiranje" površine 6
   2.4. Čišćenje i sortiranje .dat datoteke 10
   2.5. Ručno generiranje putanje 10
3. ALGORITAM ZA AUTOMATSKO GENERIRANJE TRAJEKTORIJA 13
   3.1. Sortiranje .dat datoteke 13
   3.2. Sortiranje točaka u matrici 14
   3.3. Generiranje putanje 15
      3.3.1. Matrica u excel-u 15
      3.3.2. Putanje (Curve) 16
   3.4. Nove varijable 19
   3.5. Promjena 20
4. SIMULACIJA U ROBODK-U 21
5. PRIMJERI GENERIRANIH PUTANJA 25
6. ZAKLJUČAK 28

LITERATURA 29

PRILOZI 30

POPIS SLIKA

Slika 1. Robot za poliranje ... 2
Slika 2. Brušenje zahtjevnije površine pomoću robota ... 3
Slika 3. Robot s nastavkom za brušenje ... 3
Slika 4. Robot za bojanje ... 4
Slika 5. Logo CATIA-e ... 5
Slika 6. Join alat u CATIA-i ... 6
Slika 7. Surface mesher ... 7
Slika 8. Mapped Mesh ... 8
Slika 9. "Mesh-irana" površina ... 9
Slika 10. Export Mesh ... 9
Slika 11. Logo Microsoft excel-a ... 10
Slika 12. Izgled Macro-a ... 10
Slika 13. Ispisane točke u CATIA-i ... 11
Slika 14. Putanja po točkama ... 11
Slika 15. Alat 3D curve ... 12
Slika 16. Logo Matlab-a ... 13
Slika 17. Početak excel datoteke ... 16
Slika 18. Završetak excel datoteke ... 16
Slika 19. Generirana putanja u jednom smjeru ... 17
Slika 20. Generirana putanja u drugom smjeru ... 18
Slika 21. Generirana putanja na odabranoj površini ... 18
Slika 22. Alat za mjerenje duljine u CATIA-i ... 19
Slika 23. Sučelje aplikacije ... 20
Slika 24. Izgled mape ... 20
Slika 25. Logo RoboDK-a ... 21
Slika 26. Knjižnica robota ... 22
Slika 27. Knjižnica alata ... 23
Slika 28. Curve follow project ... 24
Slika 29. Simulacija robota ... 24
Slika 30. Primjer generirane putanje I ... 25
Slika 31. Primjer generirane putanje II ... 26
Slika 32. Primjer generirane putanje III ... 26

Slika 33. Primjer generirane putanje IV 27
Slika 34. Primjer generirane putanje V 27

POPIS KRATICA

CAD - Computer-aided design

CAM – Computer-aided manufacturing

CATIA – Computer Aided Three-Dimensional Interactive Application

SAŽETAK

Ovaj rad se bavi razvojem algoritma koji će omogućiti automatsko generiranje putanje robotskog alata na temelju geometrije CAD modela.
Prvi dio rada opisuje područje primjene. Nadalje, opisuju se mogućnosti ručnog generiranja putanje alata te je napravljen uvod u razvoj algoritma popraćen njegovom detaljnom razradom. Razvijeni algoritam ispitan je na konkretnim primjerima u laboratorijskim uvjetima što je opisano na samom kraju završnog rada.

Ključne riječi: algoritam, automatsko generiranje putanje, robot, simulacija, CAD model

SUMMARY

The topic of this bachelor's thesis lays on development of an algorithm, whose main assignment is to make automated robot path planning, using CAD model.

The initial part of the work explains processes like robot polishing, robot grinding and robot painting, in which we use robot path planning.

After getting acquinted with processes, it is showed how ro make manual path planning and a little introduction into algorithm development. Following introduction, it is showed step by step on how was the algorithm developed.

After the algorithm was done, it was implemented in application which made it user friendly.

Along with the algorithm, at the end of the thesis, robot simulation is also displayed.

Key words: algorithm, automated robot path planning, robot, CAD model, simulation

1. UVOD

U mnogim robotskim primjenama, kao što su poliranje, brušenje ili bojanje, moguće je koristiti CAD model proizvoda za generiranje radnih putanja koje prate površinsku geometriju proizvoda.

Ručno generiranje takvih putanja iziskuje znatan utrošak vremena. Uz to, svaki CAD model treba različit pristup zbog geometrijskih specifičnosti.

Stoga je u ovom završnom radu prikazan razvoj algoritma za automatsko generiranje radnih putanja robota na temelju CAD modela proizvoda.

U svrhu razvoja algoritma korišteno je nekoliko softverskih alata; CATIA, Matlab, Microsoft excel, RoboDK.

1.1. Pregled područja primjene

1.1.1. Robotsko poliranje

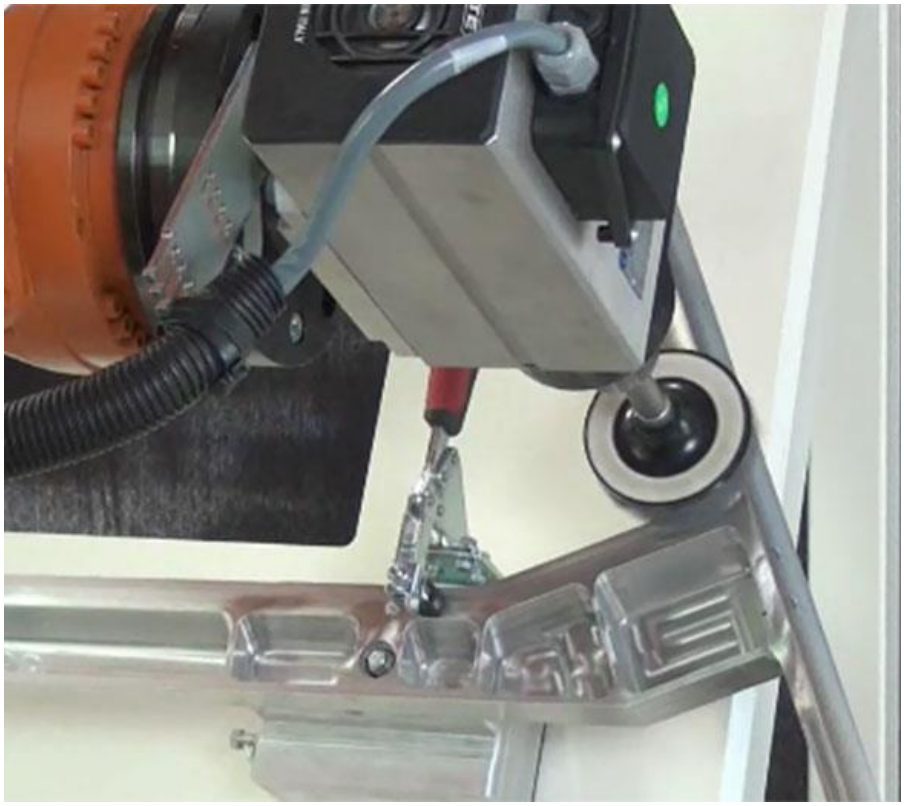
Poliranje je postupak strojne obrade skidanjem čestica, koji se koristi za poboljšanje izgleda obratka. Poliranjem se uklanja oksidacija na obratku, stvara se reflektirajuća površina, smanjuje se trenje na stjenkama cijevi. Koristi se u svrhu stvaranja ravne površine bez grešaka.

Poliranje se koristi najčešće nakon brušenja, te se njime može postići hrapavost površine N1.

Sam postupak poliranja sastoji se od trenja između obratka, alata za poliranje (relativno velike brzine) i paste za poliranje.

Postoji više vrsta poliranja, npr. ručno poliranje, automatizirano na alatnom stroju (polirka), na CNC stroju, te najznačajnije za rad, robotsko poliranje.

Na sljedećim primjeru [Slika 1.] prikazan je primjer robota kojim se polira obradak. Robot za poliranje može imati različit broj sloboda gibanja, ovisno o kompleksnosti rada.



Slika 1. Robot za poliranje

1.1.2. Robotsko brušenje

Isto kao i poliranje, brušenje je postupak strojne obrade skidanjem čestica. Brušenje se upotrebljava za završnu i finu obradu do hrapavosti površine N3.

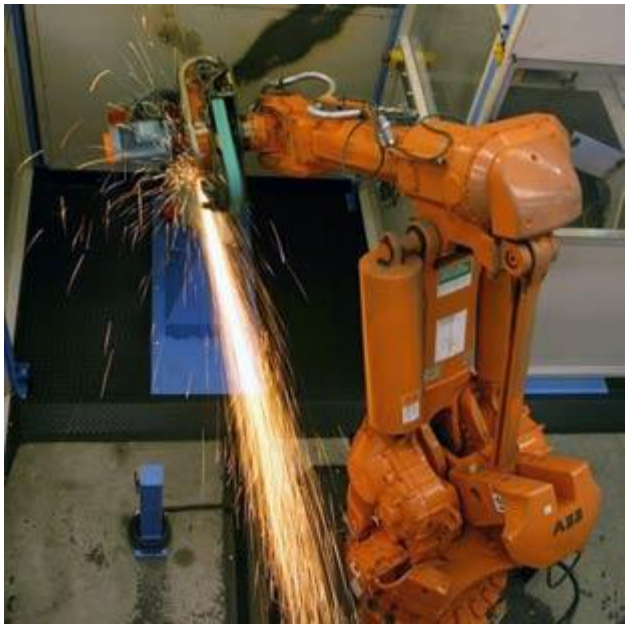
Brušenje se najčešće izvodi na brusilicama, ali može se također koristiti i robot za robotsko brušenje obradaka u slučajevima kada je u pitanju složenija površina za obradu.

Alat za brušenje je brus sastavljen od većeg broja reznih oštrica, koje se nalaze na brusnim zrnima. Za izradu brusa najčešće se korist korund (Al2O3), silicijev karbid (SiC), kubični bornitrid (CBN) i polikristični dijamant (PCD).

Brušenje se dijeli na istosmjerno i protusmjerno, te na obodno i čeono.

Slike [Slika 2., Slika 3.] prikazuju robotsko brušenje obradaka, za zahtjevnije površine.

Slika 2. Brušenje zahtjevnije površine pomoću robota



Slika 3. Robot s nastavkom za brušenje

1.1.3. Robotsko bojanje i lakiranje

Obrade u kojima se sve više koristi robot su bojanje i lakiranje.

Zbog velikog udjela štetnih i kancerogenih tvari u bojama i lakovima sve se više koristi robotsko bojanje i lakiranje što povećava humanizaciju rada.

Robotsko bojanje [slika 4.] i lakiranje povećava preciznost i ujednačenost postupka. Zbog toga se dobivaju ljepši i uredniji obratci koji prolaze kroz postupak bojanja i lakiranja.

Slika 4. Robot za bojanje

2. ANALIZA CAD MODELA

2.1. CAD model

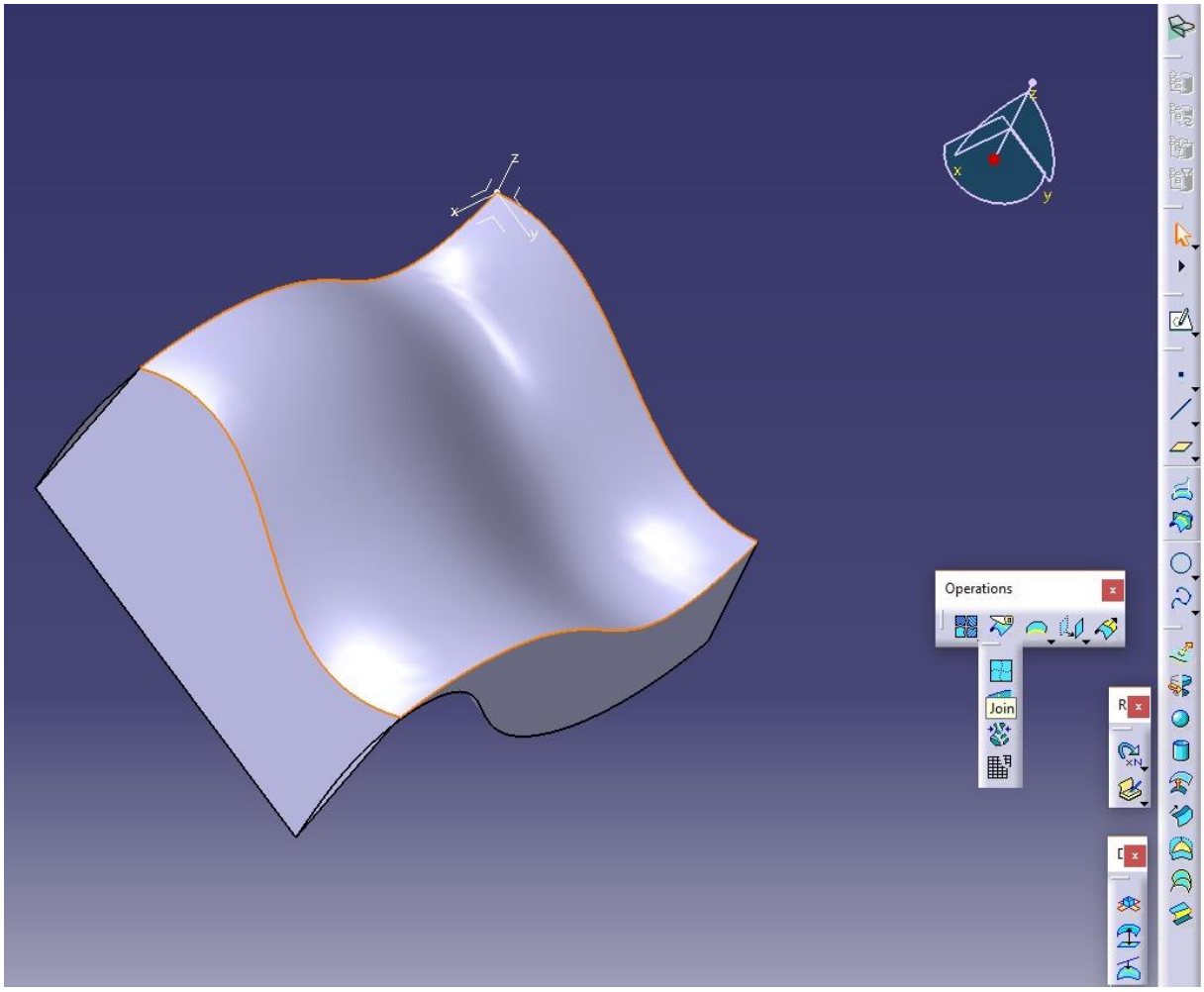
Za sam početak razvoja algoritma potreban je CAD model. CAD modeli korišteni u završnom radu samostalno su nacrtani, dobiveni su od strane asistenta ili su uzeti već gotovi modeli s interneta (GrabCAD). CAD modeli obrađivali su se u softverskom alatu CATIA.

CATIA (**Computer Aided Three-dimensional Interactive Application**) je prvi alat korišten u izradi algoritma. CATIA je CAD/CAM softverski alat u kojem se obrađivao CAD model. Inačica korištena u ovom radu je V5R20.

Slika 5. Logo CATIA-e

2.2. Površina (Surface)

Nakon odabira CAD modela sljedeći korak je odabir površine (četverostrane) na tom modelu. Željena površina je odabrana i označena pomoću alata join [slika 6.].



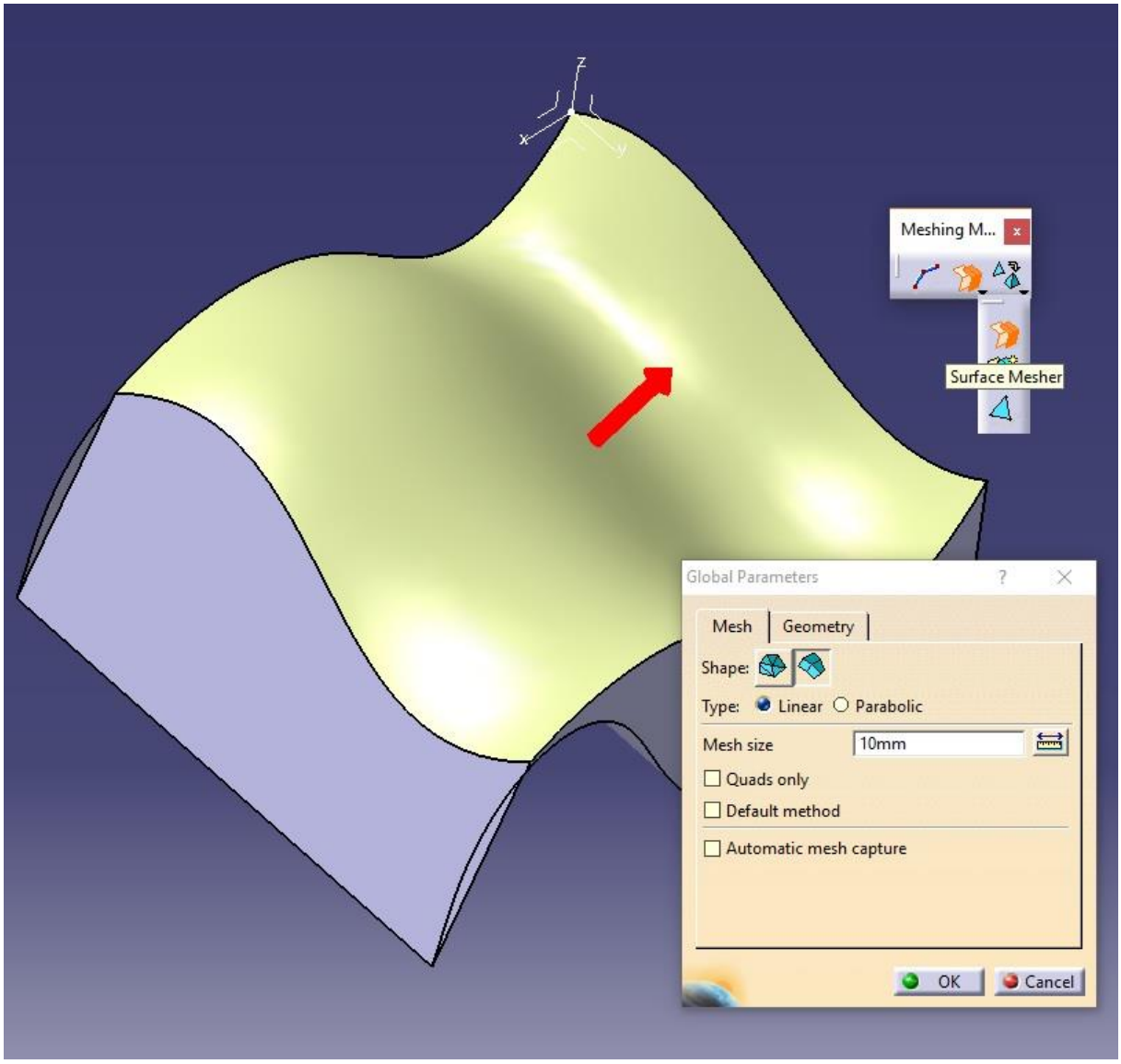
Slika 6. Join alat u CATIA-i

Join alat nalazi se u podskupu alata Wireframe and Surface design, u skupu Operations. Alatom join elementi površine spojeni su u jednu cjelinu te je površina spremna za sljedeći korak.

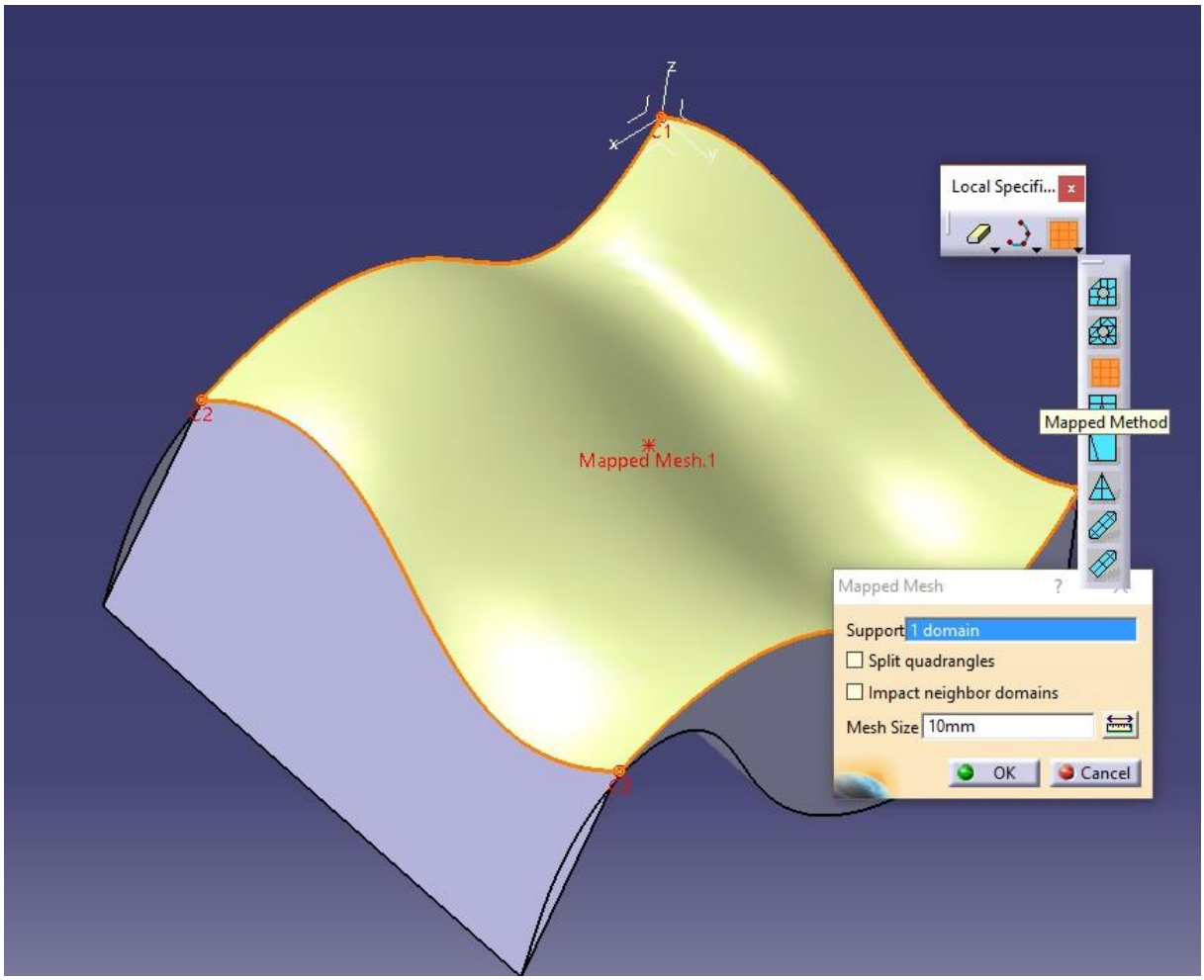
2.3. Stvaranje točaka i "meshiranje" površine

Generirana putanja na površini prolazi kroz odabrane točke na istoj.

Pomoću CATIA-e napravljen je mesh odabrane površine. Mesh-om se označava površina kojoj je struktura izrađena od konačnog broja isprepletenih elemenata. Mesh alat nalazi se u podskupu alata Advanced Meshing Tools pod nazivom Surface mesher [slika 7.]. Nakon izrade Surface mesh-a, pomoću Mapped method napravljena je mreža kvadratića na toj površini [slika 8.].

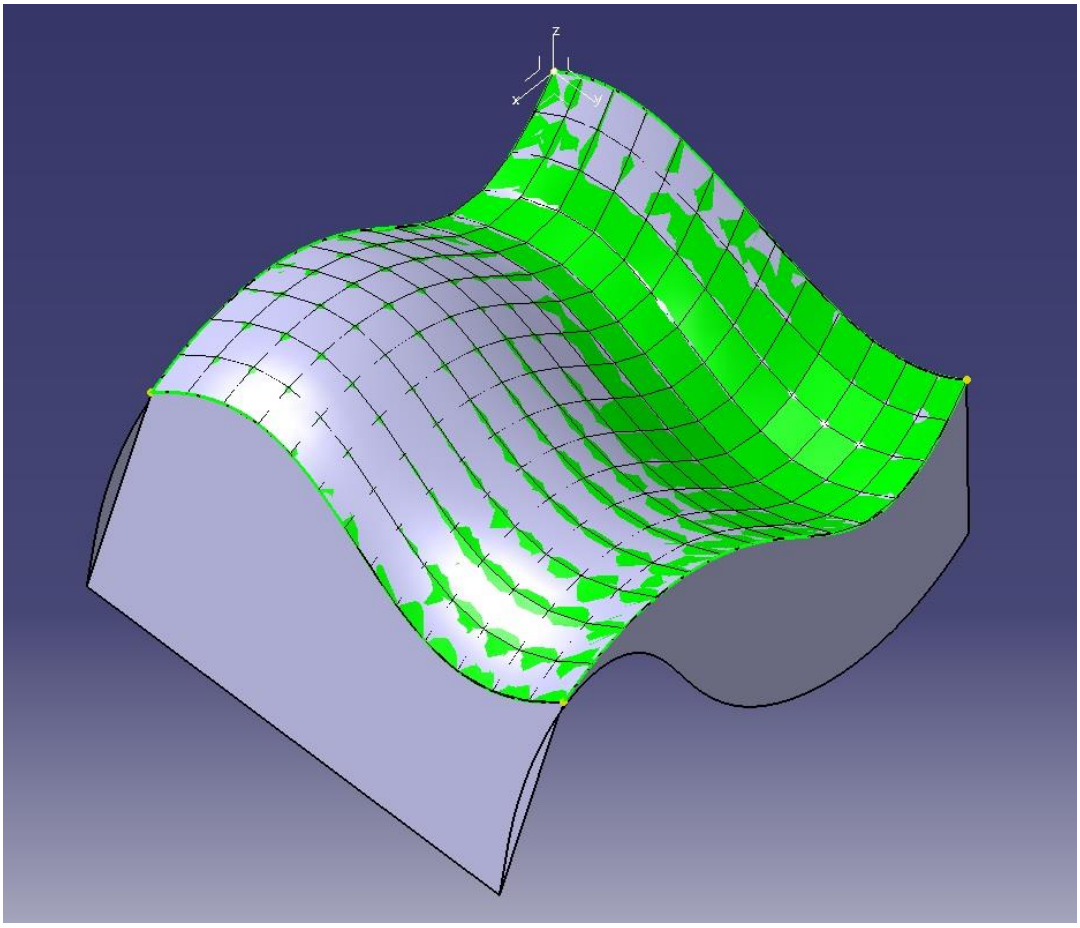


Slika 7. Surface mesher



Slika 8. Mapped Mesh

Dobivenim kvadratićima omogućena je promjena duljine stranice [slika 7. i slika 8.], te je u skladu s odabranom duljinom CATIA napravila kvadratiće sa zadanim vrijednostima [slika 9.].



Slika 9. ''Mesh-irana'' površina

Svaki kvadratić ima četiri točke, po jednu na svakom njegovom vrhu. Zbog odgovarajućeg položaja točaka, njihovi podaci pomoću export mesh-a [slika 10.] izvedeni su u .dat datoteku. Dobivena datoteka sadržava vrijednosti koordinata (x,y,z) tih točaka.

Slika 10. Export Mesh

2.4. Čišćenje i sortiranje .dat datoteke

Dobivena datoteka sadržava mnogo podataka koji su nepotrebni. Datoteka se mora pročistiti da bi se dobile samo koordinate (x,y,z) točaka.

Za čišćenje datoteke koristi se Microsoft excel i Matlab.

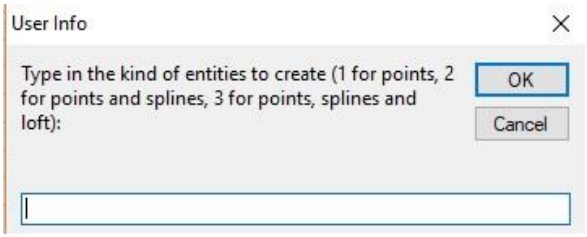
Microsof excel je dio Microsoft office paketa koji služi kao interakcija između CATIA-e i Matlab-a, te su kroz Microsoft excel pomoću Macro-a ispisivani podaci u CATIA-u.

Macro je računalni kod koji obrađuje podatke na prije zadan način kroz neki zadani uzorak.

Slika 11. Logo Microsoft excel-a

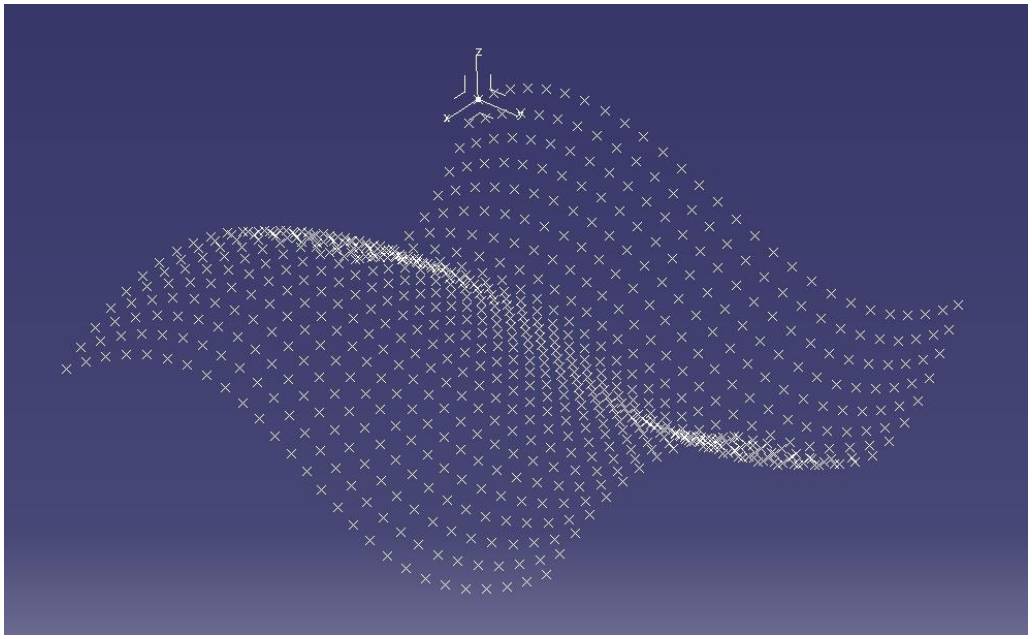
2.5. Ručno generiranje putanje

Nakon što se čišćenjem datoteke dobiju koordinate točaka u Microsoft excelu, korištena je Macro skripta [slika 12.] u excelu koja se dobije uz instaliranu verziju CATIA-e. Macro skripta radi tako da učitava podatke iz excel-a i u otvoreni „Part“ unutar CATIA-e izrađuje geometrijski set podataka. Set podataka može se sastojati od višestrukog broja točaka, krivulja i površina.



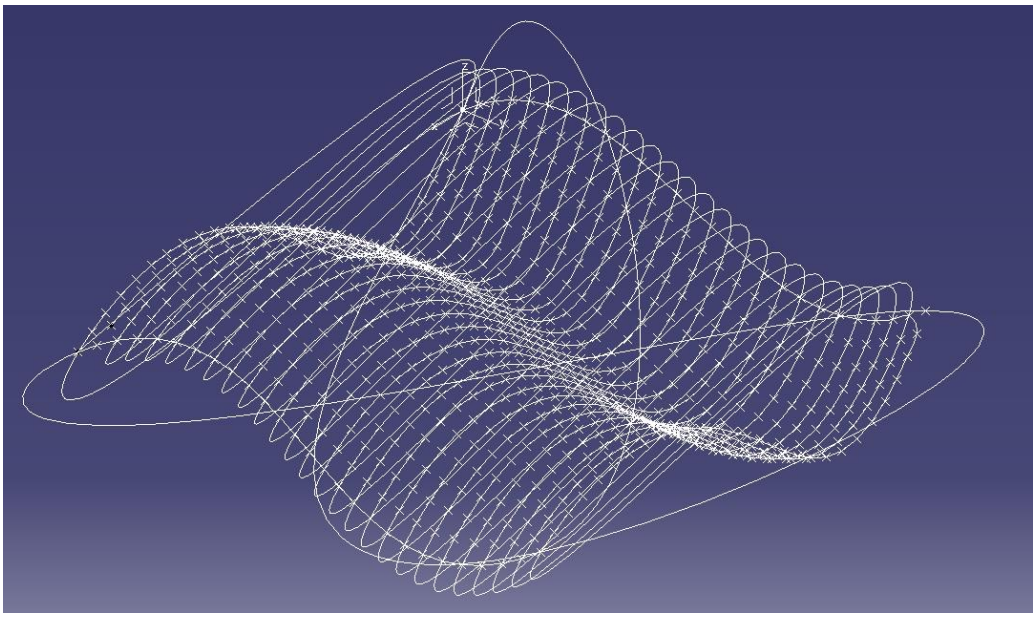
Slika 12. Izgled Macro-a

Macro skripta u CATIA-i je izradila točke [slika 13.] kojima su koordinate ispisane u Microsoft excelu.



Slika 13. Ispisane točke u CATIA-i

Skripta je nakon odabira druge varijante, uz točke napravila i putanju po istim [slika 14.]. Dobivena putanja prolazila je kroz odabrane točke, međutim redoslijed točaka nije bio u skladu s traženim redoslijedom kojim bi se dobile željene putanje.

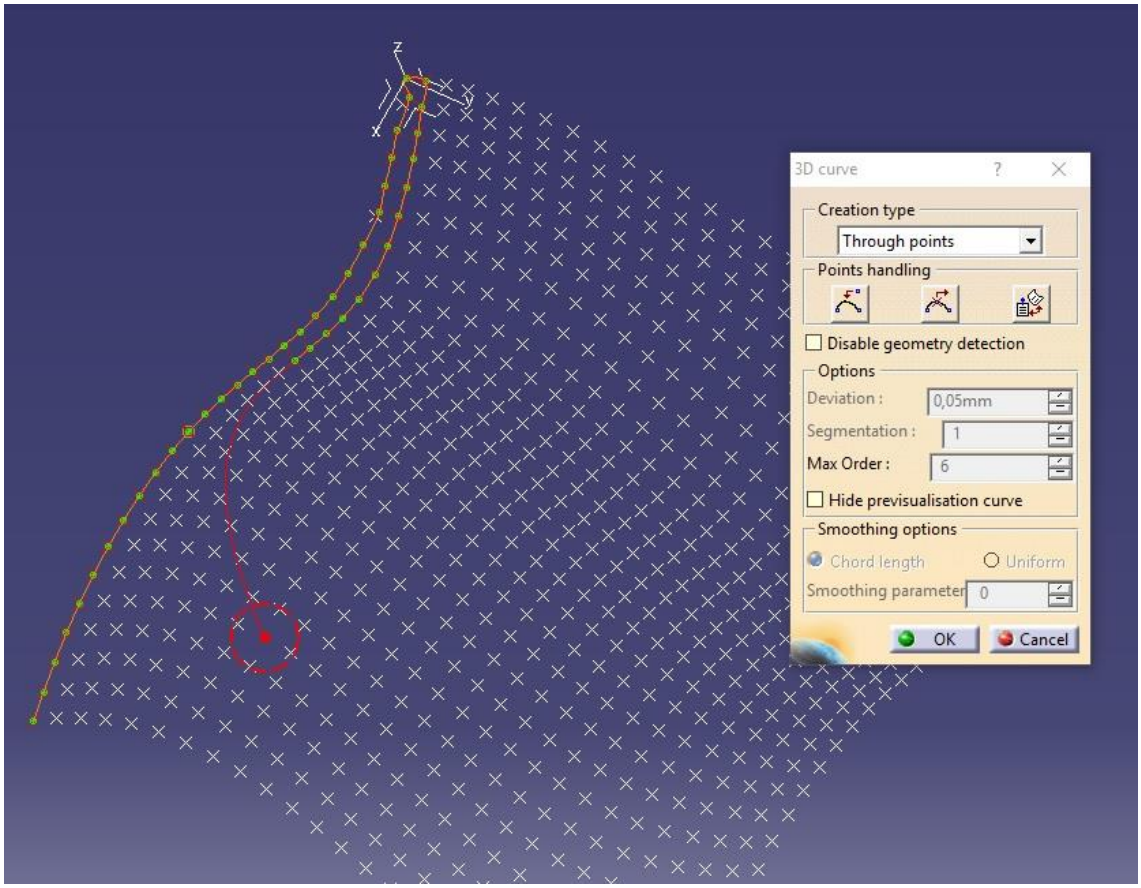


Slika 14. Putanja po točkama

Ručno generiranje putanje opisano je na dva načina.

Prvi način je premještanje koordinata ručno u Microsoft excelu. Taj proces je predugačak, a često i neprecizan, pogotovo s većim brojem točaka na odabranoj površini.

Drugi način je pomoću prije korištene Macro skripte ispisati točke u CATIA-u te koristeći alat 3D curve spojiti točke ručno u željenom redoslijedu [slika 15.]. 3D curve alat nalazi se u skupu podalata Shape-FreeStyle. Taj način je precizniji, ali vrijeme utrošeno na taj proces bilo bi predugačko, pogotovo kod velikog broja točaka.



Slika 15. Alat 3D curve

3. ALGORITAM ZA AUTOMATSKO GENERIRANJE TRAJEKTORIJA

Razvoj algoritma započeto je programiranjem, a korišteni alat je Matlab. Matlab je programski jezik koji je poslužio za matrično i numeričko računanje, te za vizualizaciju i programiranje algoritma. Matlab se pokazao kao najbolje rješenje za probleme tijekom razvijanja algoritma. Korištena verzija Matlaba je R2016a.

Slika 16. Logo Matlab-a

Prvi korak kod generiranja trajektorija - određivanje površine te izrada Mesh-a napravljen je na jednak način kao što je prikazano i opisano u prijašnjem dijelu rada (poglavlja 2.1., 2.2. i 2,3.).

3.1. Sortiranje .dat datoteke

Programiranje u Matlabu započinje ubacivanjem podataka iz .dat datoteke u Matlab te njeno sortiranje i čišćenje.

```
>> str = fileread('ime_dat_datoteke.dat');
```

Funkcija fileread učitala je odabranu datoteku.

```
>> tkn = regexp(str,'^GRID\*\s+\d+\s+([+-]?\d+,\d+)\s+([+-]?\d+,\d+)\s+\d+\s+\*\s+\d+\s+([+-]?\d+,\d+)',
                    'lineanchors','tokens');
tkn = vertcat(tkn{:});
tkn = strrep(tkn,',','.');
mat = str2double(tkn);
```

Funkcijom regexp omogućeno je traženje i zamjena određenog teksta kroz cijelu datoteku, dok je str funkcijom tekst pretvoren u string.

Funkcijom vertcat napravljen je vertikalni niz od podataka koji su ostali nakon korištenja funkcije regexp, dok je strrep funkcijom dobivena kopija istog vertikalnog niza.

Funkcijom str2double string je pretvoren u double.

Sortiranjem se dobije očišćena datoteka, tj. matrica koja se sastoji od tri stupca. Svaki od stupaca označuje jednu koordinatnu os(x,y,z).

Dobivena matrica može se ispisati u Microsoft excel datoteku pomoću funkcije xlswrite.

3.2. Sortiranje točaka u matrici

Željeni algoritam mora dobivenu matricu soritrati na način da se dobije odgovarajući raspored točaka. Izvršena su dva načina sortiranja točaka; prvi je način sortiranja točaka okomito po odabranoj površini, dok je drugi sortiranje točaka vertikalno.

```
>> Y=M-2;
U=2*M+2*N-4;
V=U+1;
A=mat(V:end,1:3);
[e,f]=size(A)
E=e/(2*Y)
for k=0:E;
A(k*(2*Y)+(Y+1):k*(2*Y)+(2*Y),1:3)=A(k*(2*Y)+(2*Y):-1:k*(2*Y)+(Y+1),1:3);
end;
```

Prvi način sortiranja točaka okomito, započinje upisivanjem varijabli: M; koja označava broj točaka na jednoj stranici površine, N; koja označava broj točaka na drugoj stranici površine.

Dobivena vrijednost U označava broj točaka koje se izbacuje iz matrice, dok je vrijednost V broj točke do koje treba izbrisati nepotrebne koordinate. Varijabla E govori koliko skupova točaka ima na površini. Jedan skup točaka poprima dvostruki broj točaka koji se nalaze na jednoj stranici. Na kraju, for petlja sortira matricu te se dobije konačna matrica A.

```
>> Y=M-2;
Z=N-2;
U=2*M+2*N-4;
V=U+1;
A=mat(V:end,1:3);
B=A;
S = size(B);
B = reshape(permute(reshape(B,Y,[],S(2)),[2,1,3]),S);
[e,f]=size(A);
E=e/(2*Y);
for j=0:E;
B(j*(2*Z)+(Z+1):j*(2*Z)+(2*Z),1:3)=B(j*(2*Z)+(2*Z):-1:j*(2*Z)+(Z+1),1:3);
end;
```

Drugi način sortiranja točaka vodoravno, započinje kao i prvi način. Promjena koja se javlja je izmjena redova u matrici A, što rezultira dobivanjem matrice B. Dobivena matrica B ima koordinate ispisane u vodoravnom redoslijedu. Funkcija koja obavlja tu promjenu je reshape.

3.3. Generiranje putanje

Dobivene matrice imaju željeni raspored točaka. Matrice se izvedu u Microsoft excel datoteke te se Macro-om naprave putanje tražene od samog početka. Macro skripta ista je kao u poglavlju 2.5.

3.3.1. Matrica u excel-u

Matrica je izvedena iz Matlaba u Microsoft excel pomoću funkcije xlswrite. U excel datoteku odmah se upisuju i potrebne linije za početak Macro-a.

```
>> xlswrite('ime_excel_file.xlsx', [{'''StartLoft', [],[];'''StartCurve', [],[]};num2cell(A);
            {'''EndCurve', [],[];'''EndLoft', [],[];'''End',[],[]}]);
xlswrite('ime_excel_file.xlsx', [{'''StartLoft', [],[];'''StartCurve', [],[]};num2cell(B);
            {'''EndCurve', [],[];'''EndLoft', [],[];'''End',[],[]}]);
```

Dobivene matrice imaju početne linije koda [slika 17.] ; StartLoft i StartCurve, zatim su ispisane koordinate točaka i na kraju su ispisane linije koda [slika 18.] ; EndCurve, EndLoft i End.

	A	B	C
1	StartLoft		
2	StartCurve		
3	2,850472	115,9492	-6,94244
4	5,948829	115,9331	-10,8683
5	9,355684	115,906	-14,5297
6	13,15053	115,8722	-17,7834
7	17,40628	115,8107	-20,3996
8	22,12447	115,7353	-22,0241
9	27,09739	115,6731	-22,3152
10	31,97125	115,6274	-21,2658
11	36,53042	115,6059	-19,2241
12	40,782	115,5929	-16,5928
13	44,82138	115,5874	-13,6445

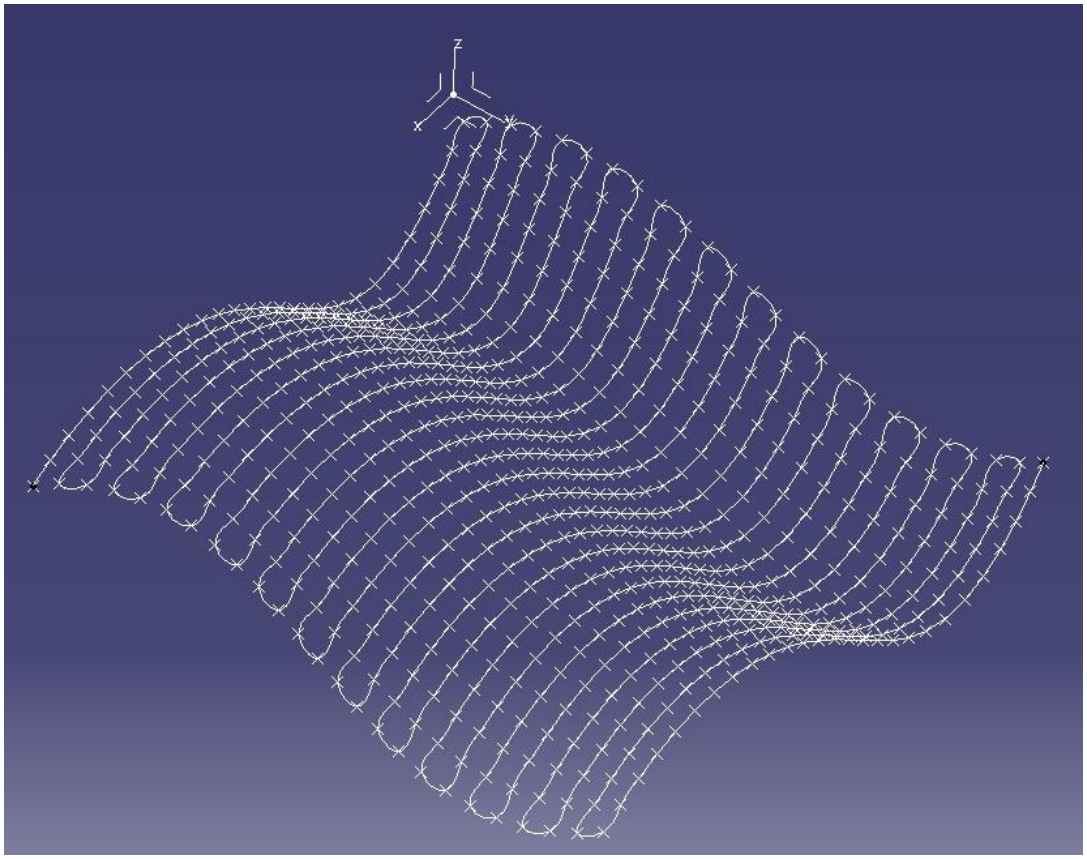
Slika 17. Početak excel datoteke

669	76,2546	4,350468	9,705404
670	81,07618	4,330069	10,85242
671	85,9884	4,300092	11,50278
672	90,94003	4,270497	11,64537
673	95,88058	4,235703	11,27742
674	100,7584	4,194707	10,40999
675	105,5295	4,156856	9,070268
676	110,1551	4,116714	7,292086
677	114,609	4,07711	5,119939
678	EndCurve		
679	EndLoft		
680	End		

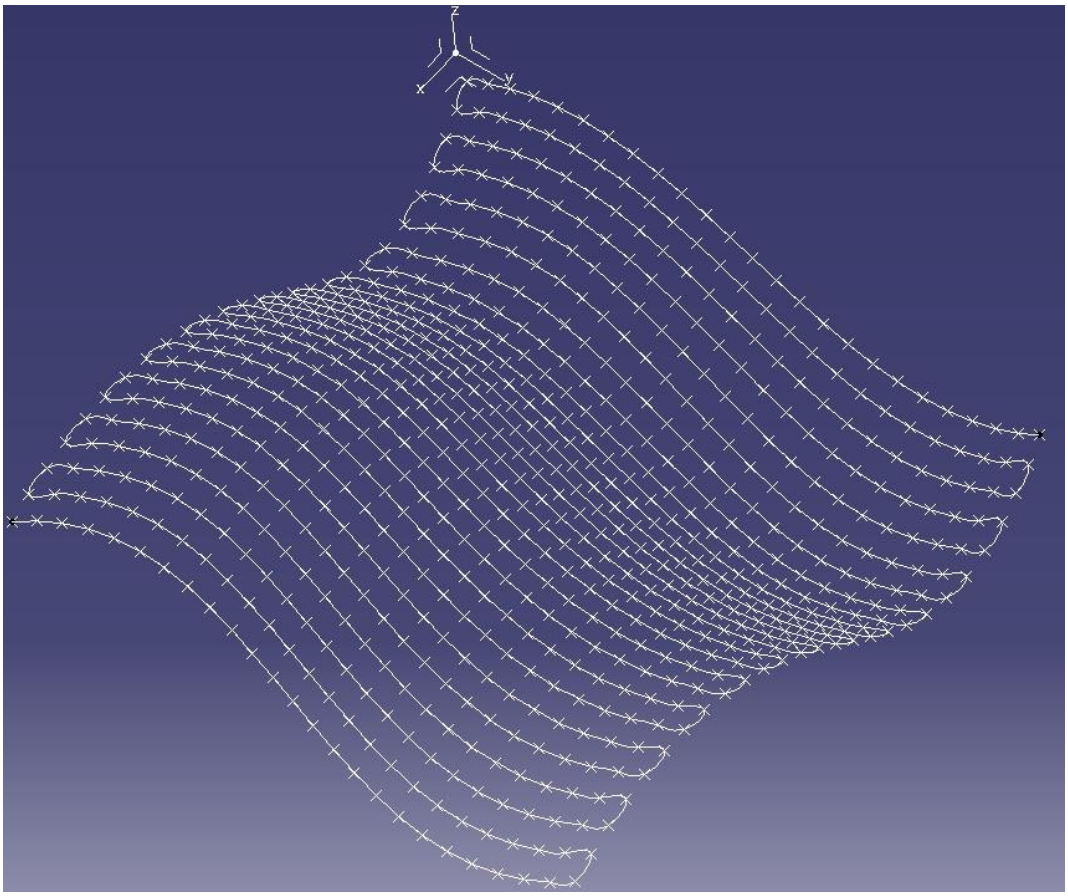
Slika 18. Završetak excel datoteke

3.3.2. Putanje (Curves)

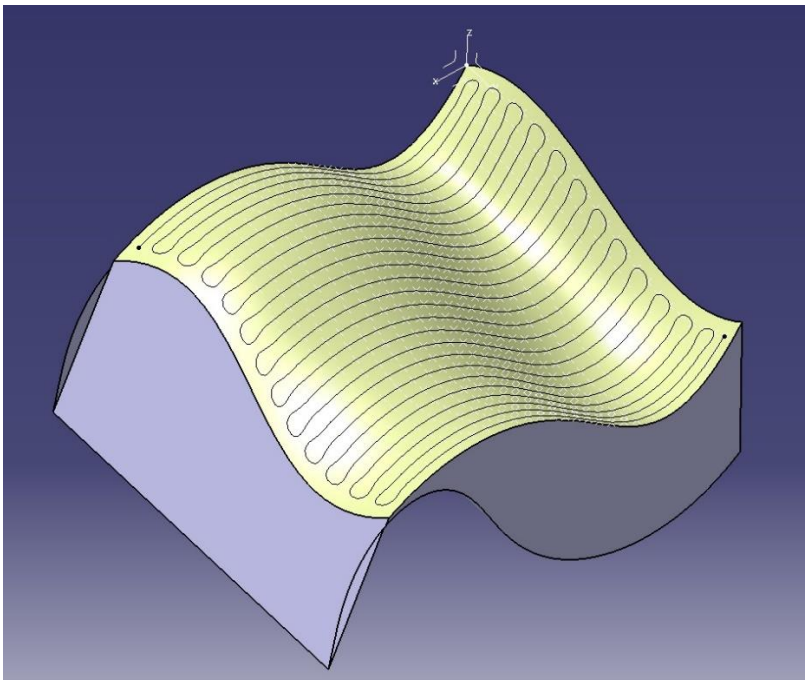
U dobivenim excel datotekama pokrenuta je Macro skripta. Skripta je obradila ispisane koordinate te su dobivene konačne putanje po odabranoj površini. Dobivena je putanju u jednom smjeru [slika 19.] te putanja u drugom smjeru [slika 20.]. Slika 21. prikazuje generiranu putanju implementiranu na površinu po kojoj je ta putanja napravljena.



Slika 19. Generirana putanja u jednom smjeru



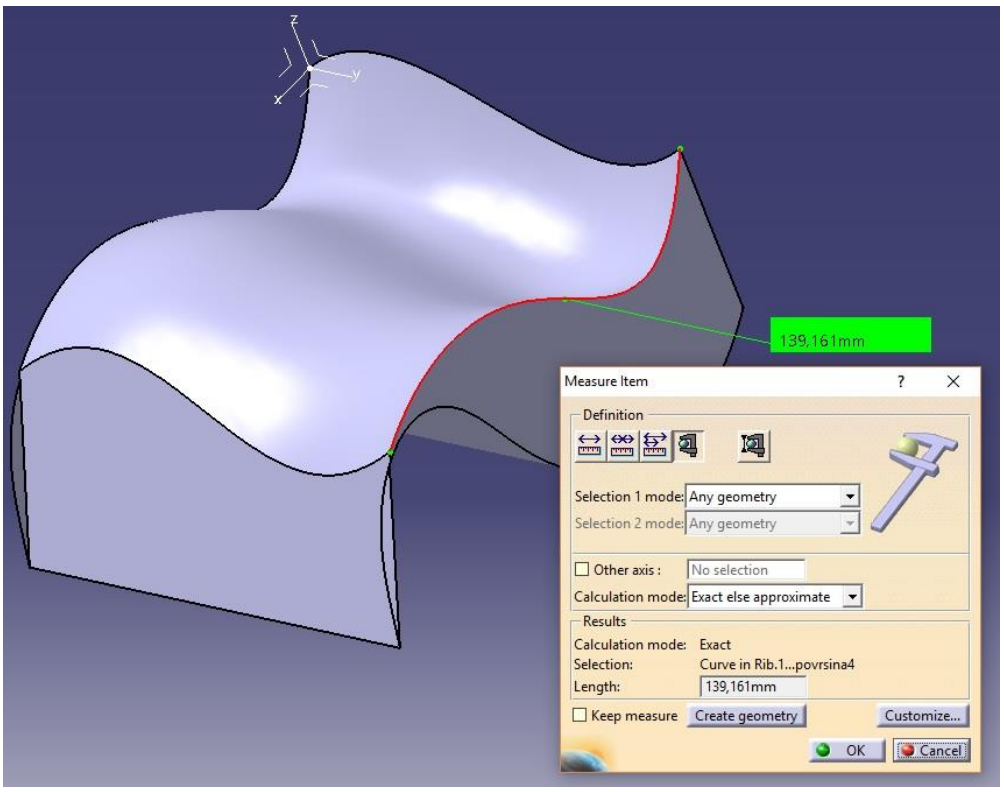
Slika 20. Generirana putanja u drugom smjeru



Slika 21. Generirana putanja na odabranoj površini

3.4. Nove varijable

U dobivenom algoritmu varijable su broj točaka na duljini i širini površine, veličina mesha te ukupan broj točaka. Korisnik koji bi koristio algoritam teško bi dolazio do tih podataka, zbog toga su potrebne promjene varijabli kako bi se podaci prilagodili korisnicima koji će moći s lakoćom iščitati iste u CATIA-i [slika 22.]. Nove varijable su duljina stranica te veličina mesh-a. Duljina stranica izmjeri se alatom Measure, dok je veličina mesh-a odabrana kod izrade mesha površine (poglavlje 2.3.) .



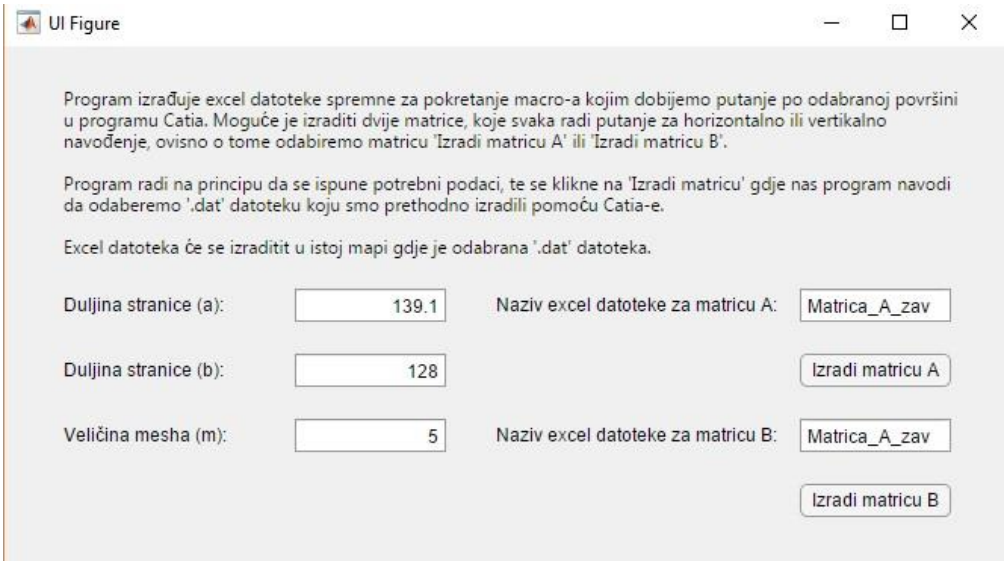
Slika 22. Alat za mjerenje duljine u CATIA-i

Pomoću novih varijabli dobivaju se vrijednosti varijabli prethodno potrebnih za algoritam.

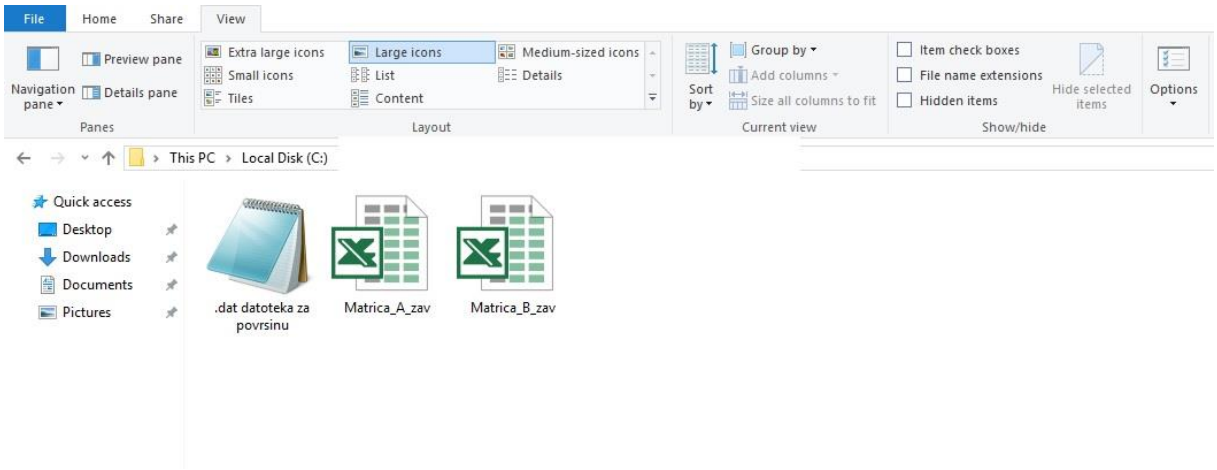
```
>> a= ;   %duljina jedne stranice povrsine
b= ;   %duljina druge stranice povrsine
m= ;   %velicina mesha
M=a/m;
M=round(M);
M=M+1;   %M je broj tocaka na jednoj stranici povrsine
N=b/m;
N=round(N);
N=N+1;   %N je broj tocaka na drugoj stranici povrsine
```

3.5. Promjena

Zbog lakše preglednosti i jednostavnosti, ali i očuvanja vlastitog koda, napravljena je aplikacija [slika 23.] u kojoj je izveden algoritam. Aplikacija je jednostavna za korištenje. U nju se jednostavno upiše duljina stranica, veličina mesha te željeni naziv excel datoteke u koju će se ispisati tražena matrica. Aplikacija sama pita koju .dat datoteku korisnik želi odabrati te u istoj mapi gdje se nalazi datoteka sprema excel datoteke s traženim koordinatama točaka [slika 24.]. Mogu se dobiti dvije matrice, ovisno o smjeru (okomito ili vodoravno) koji je potreban za željenu putanju.



Slika 23. Sučelje aplikacije



Slika 24. Izgled mape

4. SIMULACIJA U ROBODK-U

Završni dio rada prikazuje simulaciju robota u softverskom programu RoboDK.

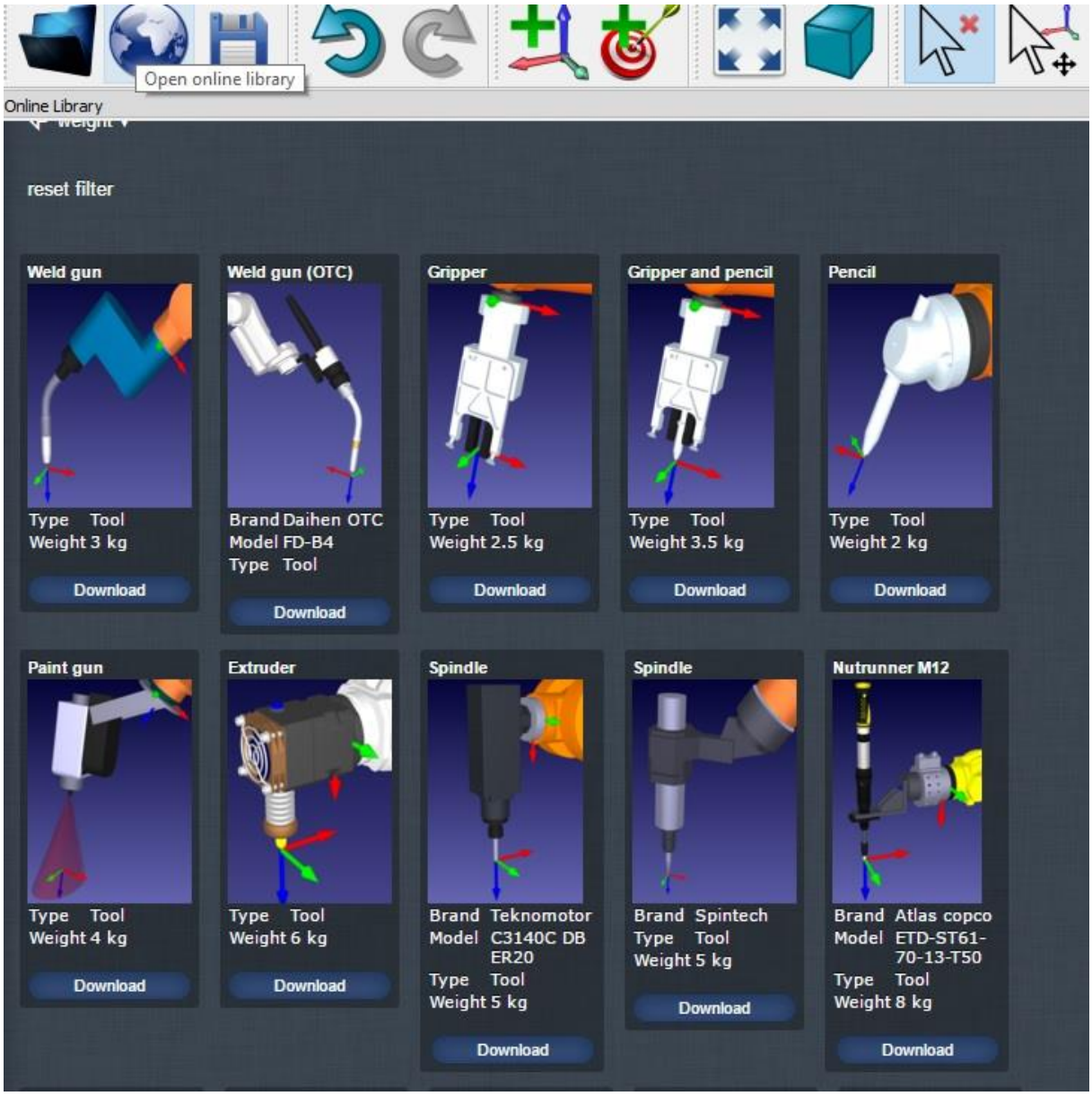
RoboDK je softverski alat koji služi za programiranje i simulaciju industrijskih robota.

Slika 25. Logo RoboDK-a

Simulacija prikazuje alat robota koji se pomiče po putanji dobivenoj algoritmom.

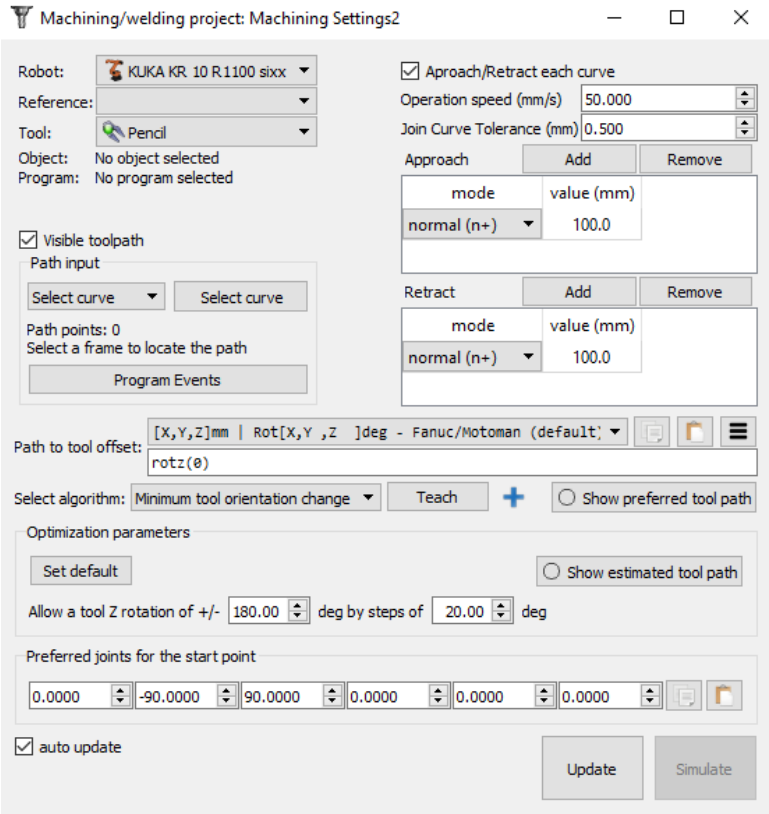
Rad započinje odabirom robota u knjižnici robota ponuđen od strane RoboDK-a [slika 26.]. U istoj toj knjižnici ponuđeni su također i alati za robote [slika 27.], gdje se bira alat potreban za robotsku radnju. Roboti i alati mogu se isto tako dodavati i sami ukoliko je potreban alat koji se ne nalazi među ponuđenim alatima u navedenoj RoboDK knjižnici.

Slika 26. Knjižnica robota

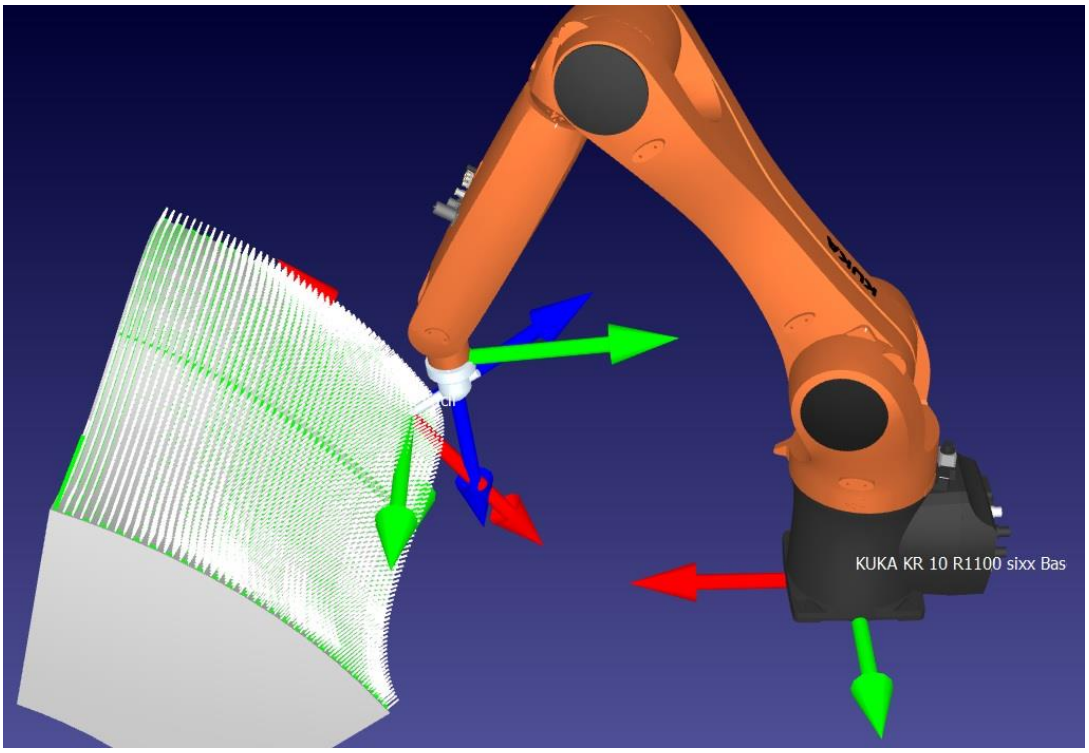


Slika 27. Knjižnica alata

Sljedeći korak je otvaranje željenog obratka koji na sebi ima generiranu putanju. Nakon toga namješta se položaj robota, alata i obratka te pomoću operacije Curve follow project [slika 28.] izrađuje se simulacija [slika 29.]. U samom prozoru odabrane operacije odabire se robot i alat kojim se izvršava određena radnja te sama putanja koju robot prati. Mogu se odabrati i željeni prilazak robota obratku te odmicanje od robota, kao i brzina robota.



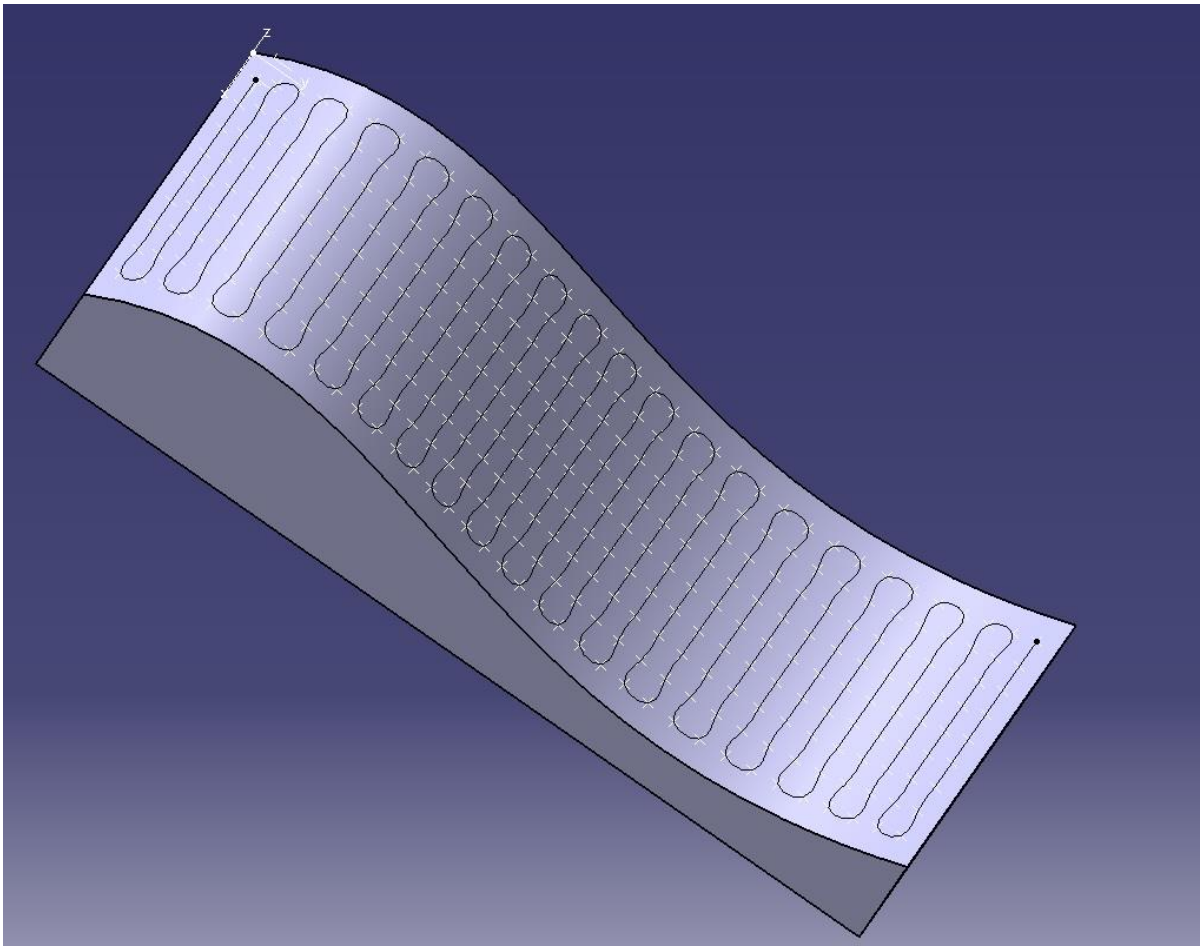
Slika 28. Curve follow project



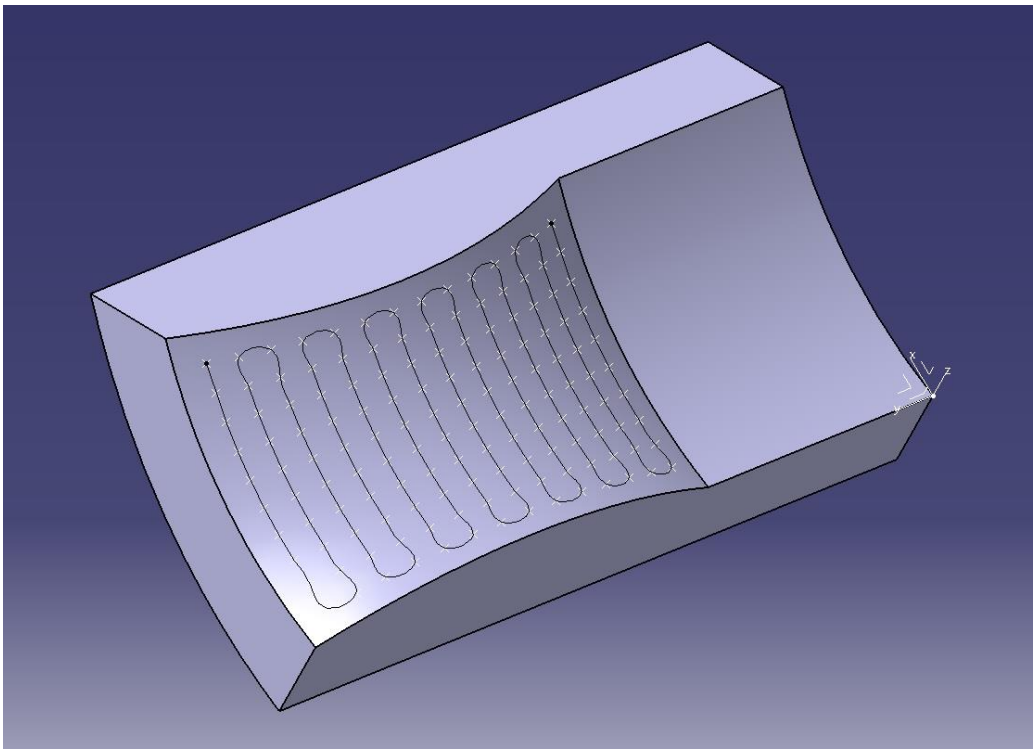
Slika 29. Simulacija robota

5. PRIMJERI GENERIRANIH PUTANJA

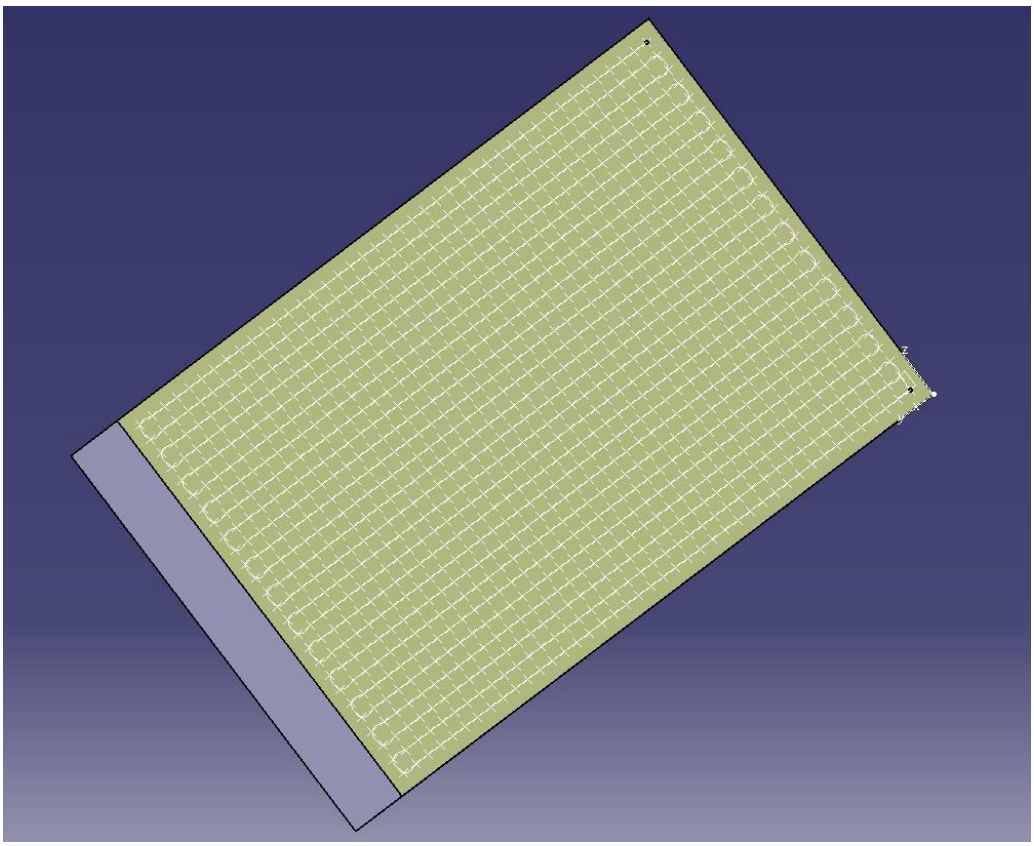
Da bi se dobio bolji uvid u rad algoritma prikazani su primjeri raznih putanja na različitim površinama. Svaka od tih putanja generirana je pomoću algoritma te simulirana u RoboDK-u.



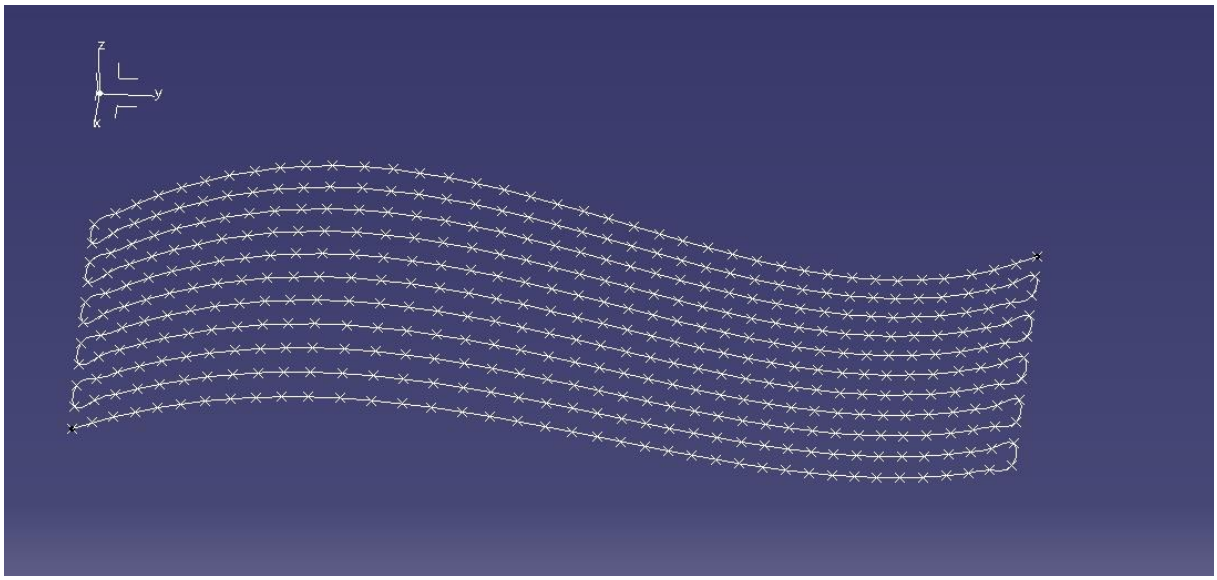
Slika 30. Primjer generirane putanje I



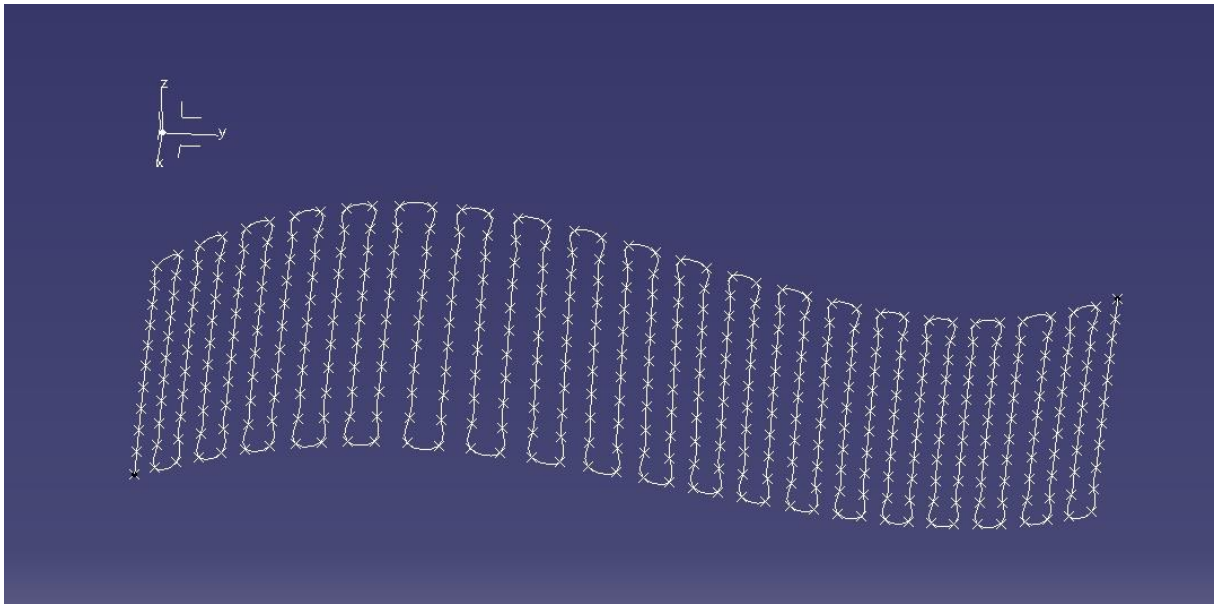
Slika 31. Primjer generirane putanje II



Slika 32. Primjer generirane putanje III



Slika 33. Primjer generirane putanje IV



Slika 34. Primjer generirane putanje V

6. ZAKLJUČAK

U današnje vrijeme korištenje robota se sve više povećava i nalaze se novi načini na koje bi se implementirale mogućnosti robota. Jedan od mnogih zadataka koji robot može obavljati upravo je kretanje po nekoj zadanoj putanji i obavljanje zadanog zadatka. Na taj način, robot može asistirati u obradama kao što su brušenje, poliranje, bojanje, lakiranje i slično.

Kreiranje putanje kojom se kreće robotski alat posao je koji oduzima velik dio radnog vremena. Algoritam koji je opisan u ovom završnom radu, može uvelike pomoći i ubrzati proces generiranja tražene putanje. Kako bi bilo što jednostavnije korištenje napravljenog algoritma, objedinjen je u aplikaciji kojom korisnik može lako rukovati.

Algoritmom se dobivaju podaci kojima se, uz pomoć CAD/CAM softverskog alata, izrađuju putanje. Generirane putanje programiraju se u robot, a prije toga mogu se provjeriti u softverskom alatu za simulacije robotskih sustava.

U budućnosti, primjena robota bit će sve veća i pronalazit će se novi, bolji i jednostavniji načini generiranja putanja po CAD modelu. Ovaj algoritam je samo još jedan korak dalje do potpune automatizacije proizvodnje.

LITERATURA

1. Ciglar, D. 2018. *Postupci završne obrade*, FSB prezentacija, Zagreb, https://www.fsb.unizg.hr/kas/ODIOO/ODIOO/Postupcizavrsneobrade.pdf
2. Škorić, S. 2018. *Brušenje,* FSB prezentacija, Zagreb, https://www.fsb.unizg.hr/kas/ODIOO/Brusenje_ooc.pdf
3. Robotakon, http://www.robotakton.net/meni4.php (pristupljeno u kolovozu 2018.)
4. Hunor, http://www.hunor.hr/index.php (pristupljeno u kolovozu 2018.)
5. GrabCad, www.grabcad.com (pristupljeno u lipnju 2018.)
6. 3Dcadforum, https://www.3dcadforums.com/forum.php (pristupljeno u lipnju 2018.)
7. Youth4work, https://www.youth4work.com/ (pristupljeno u srpnju 2018.)
8. Mcadcentral, https://www.mcadcentral.com/forum.php (pristupljeno u lipnju 2018.)
9. Eng-tips, https://www.eng-tips.com/ (pristupljeno u lipnju 2018.)
10. Mathworks, https://uk.mathworks.com (pristupljeno u lipnju 2018.)
11. Stackoverflow, https://stackoverflow.com/ (pristupljeno u lipnju 2018.)
12. RoboDK, https://robodk.com/ (pristupljeno u srpnju 2018.)
13. Jokić A., Essert M., Milić V. 2017. Aproksimacija i interpolacija u MATLAB-u, FSB prezentacija, Zagreb, https://www.fsb.unizg.hr/usb_frontend/files/1461834633-0-aproksimacijaiinterpolacija.pdf
14. Jokić A., Essert M., Milić V. 2017. Optimiranje u MATLAB-u, FSB prezentacija, Zagreb, https://www.fsb.unizg.hr/usb_frontend/files/1464857926-0-optimiranje_prezentacija.pdf

PRILOZI

I. CD-R disc

II. Programski kod algoritma

III. Programski kod aplikacije

IV. Aplikacija

II. PROGRAMSKI KOD ALGORITMA

```
a= ;   %duljina jedne stranice povrsine
b= ;   %duljina druge stranice povrsine
m= ;   %velicina mesha
str = fileread('ime_dat_file.dat');   %ucitava .dat file
tkn = regexp(str,'^GRID\*\s+\d+\s+([+-]?\d+,\d+)\s+([+-
]?\d+,\d+)\s+\d+\s+\*\s+\d+\s+([+-]?\d+,\d+)','lineanchors','tokens');
%sortira
tkn = vertcat(tkn{:});
tkn = strrep(tkn,',','.');
mat = str2double(tkn);   %konacna matrica iz .dat file
M=a/m;
M=round(M);
M=M+1;   %M je broj tocaka na jednoj stranici povrsine
N=b/m;
N=round(N);
N=N+1;   %N je broj tocaka na drugoj stranici povrsine
Y=M-2;   %Y je broj tocaka jedne linije
Z=N-2;   %Z je broj tocaka druge linije
U=2*M+2*N-4;   %U je broj tocaka vanjskih rubova potrebnih izbrisati iz
pocetne matrice
V=U+1;   %V broj tocke od koje treba izbrisati u matrici
A=mat(V:end,1:3);   %A je matrica bez vanjskih tocaka
B=A;
S = size(B);   %velicina matrice B
B = reshape(permute(reshape(B,Y,[],S(2)),[2,1,3]),S);   %premjestanje
tocaka za drugi smjer
[e,f]=size(A)
E=e/(2*Y)

for k=0:E;
A(k*(2*Y)+(Y+1):k*(2*Y)+(2*Y),1:3)=A(k*(2*Y)+(2*Y):-1:k*(2*Y)+(Y+1),1:3);
end;   %dobijemo konacnu matricu A

for j=0:E;
B(j*(2*Z)+(Z+1):j*(2*Z)+(2*Z),1:3)=B(j*(2*Z)+(2*Z):-1:j*(2*Z)+(Z+1),1:3);
end;   %dobijemo konacnu matricu B
```

```
xlswrite('ime_excel_file.xlsx', [{'''StartLoft', [],[];'''StartCurve',
[],[]};num2cell(A);{'''EndCurve', [],[];'''EndLoft',
[],[];'''End',[],[]}]);   %Ispisemo konacnu matricu A u excel
xlswrite('ime_excel_file.xlsx', [{'''StartLoft', [],[];'''StartCurve',
[],[]};num2cell(B);{'''EndCurve', [],[];'''EndLoft',
[],[];'''End',[],[]}]);   %Ispisemo konacnu matricu B u excel
```

III. PROGRAMSKI KOD APLIKACIJE

```
classdef Aplikacija_za_putanje < matlab.apps.AppBase

    % Properties that correspond to app components

    properties (Access = public)

        UIFigure               matlab.ui.Figure                   % UI Figure

        LabelNumericEditField  matlab.ui.control.Label            % Duljina...

        size_a                 matlab.ui.control.NumericEditField % [-Inf Inf]

        Label                  matlab.ui.control.Label            % Duljina...

        size_b                 matlab.ui.control.NumericEditField % [-Inf Inf]

        LabelNumericEditField2 matlab.ui.control.Label            % Veličin...

        mesh_size              matlab.ui.control.NumericEditField % [-Inf Inf]

        LabelEditField         matlab.ui.control.Label            % Naziv e...

        excel_first            matlab.ui.control.EditField

        Label2                 matlab.ui.control.Label            % Naziv e...

        excel_second           matlab.ui.control.EditField

        button_matrica_a       matlab.ui.control.Button           % Izradi ...

        button_matrica_b       matlab.ui.control.Button           % Izradi ...

        Label3                 matlab.ui.control.Label            % Program...

    end

    methods (Access = private)

        % Code that executes after component creation

        function startupFcn(app)
```

```
        end

        % button_matrica_a button pushed function

        function button_matrica_aButtonPushed(app)

            a=app.size_a.Value;

            b=app.size_b.Value;

            m=app.mesh_size.Value;

            [file,path] = uigetfile('*.dat');

            str = fileread(fullfile(path,file));   %ucitava .dat file

tkn = regexp(str,'^GRID\*\s+\d+\s+([+-]?\d+,\d+)\s+([+-
]?\d+,\d+)\s+\d+\s+\*\s+\d+\s+([+-]?\d+,\d+)','lineanchors','tokens');
%sortira

tkn = vertcat(tkn{:});

tkn = strrep(tkn,',','.');

mat = str2double(tkn);   %konacna matrica iz .dat file

M=a/m;

M=round(M);

M=M+1;   %M je broj tocaka na jednoj stranici povrsine

N=b/m;

N=round(N);

N=N+1;   %N je broj tocaka na drugoj stranici povrsine

Y=M-2;   %Y je broj tocaka jedne linije

Z=N-2;   %Z je broj tocaka druge linije
```

```
U=2*M+2*N-4;   %U je broj tocaka vanjskih rubova potrebnih izbrisati iz pocetne
matrice

V=U+1;   %V broj tocke od koje treba izbrisati u matrici

A=mat(V:end,1:3);   %A je matrica bez vanjskih tocaka

[e,f]=size(A);

E=e/(2*Y)-1;

            for k=0:E;

A(k*(2*Y)+(Y+1):k*(2*Y)+(2*Y),1:3)=A(k*(2*Y)+(2*Y):-1:k*(2*Y)+(Y+1),1:3);

end;

            name1=app.excel_first.Value

             xlswrite(fullfile(path,name1), [{'''StartLoft',
[],[];'''StartCurve', [],[]};num2cell(A);{'''EndCurve', [],[];'''EndLoft',
[],[];'''End',[],[]}]);   %Ispisemo konacnu matricu A u excel

        end

        % button_matrica_b button pushed function

        function button_matrica_bButtonPushed(app)

            a=app.size_a.Value;

            b=app.size_b.Value;

            m=app.mesh_size.Value;

            [file,path] = uigetfile('*.dat');

            str = fileread(fullfile(path,file));   %ucitava .dat file

tkn = regexp(str,'^GRID\*\s+\d+\s+([+-]?\d+,\d+)\s+([+-
]?\d+,\d+)\s+\d+\s+\*\s+\d+\s+([+-]?\d+,\d+)','lineanchors','tokens');
%sortira
```

```
tkn = vertcat(tkn{:});

tkn = strrep(tkn,',','.');

mat = str2double(tkn);   %konacna matrica iz .dat file

M=a/m;

M=round(M);

M=M+1;   %M je broj tocaka na jednoj stranici povrsine

N=b/m;

N=round(N);

N=N+1;   %N je broj tocaka na drugoj stranici povrsine

Y=M-2;   %Y je broj tocaka jedne linije

Z=N-2;   %Z je broj tocaka druge linije

U=2*M+2*N-4;   %U je broj tocaka vanjskih rubova potrebnih izbrisati iz pocetne
matrice

V=U+1;   %V broj tocke od koje treba izbrisati u matrici

A=mat(V:end,1:3);   %A je matrica bez vanjskih tocaka

B=A;

S = size(B);   %velicina matrice B

B = reshape(permute(reshape(B,Y,[],S(2)),[2,1,3]),S);   %premjestanje tocaka za
drugi smjer

[e,f]=size(B);

F=e/(2*Z)-1;

            for k=0:F;

B(k*(2*Z)+(Z+1):k*(2*Z)+(2*Z),1:3)=B(k*(2*Z)+(2*Z):-1:k*(2*Z)+(Z+1),1:3);
```

```
end;

            name2=app.excel_second.Value

             xlswrite(fullfile(path,name2), [{'''StartLoft',
[],[];'''StartCurve', [],[]};num2cell(B);{'''EndCurve', [],[];'''EndLoft',
[],[];'''End',[],[]}]);   %Ispisemo konacnu matricu B u excel

        end

    end

    % App initialization and construction

    methods (Access = private)

        % Create UIFigure and components

        function createComponents(app)

            % Create UIFigure

            app.UIFigure = uifigure;

            app.UIFigure.Position = [100 100 661 341];

            app.UIFigure.Name = 'UI Figure';

            setAutoResize(app, app.UIFigure, true)

            % Create LabelNumericEditField

            app.LabelNumericEditField = uilabel(app.UIFigure);

            app.LabelNumericEditField.HorizontalAlignment = 'right';

            app.LabelNumericEditField.Position = [39 163 106 15];

            app.LabelNumericEditField.Text = 'Duljina stranice (a):';

            % Create size_a

            app.size_a = uieditfield(app.UIFigure, 'numeric');
```

```
app.size_a.Position = [192 159 100 22];

% Create Label

app.Label = uilabel(app.UIFigure);

app.Label.HorizontalAlignment = 'right';

app.Label.Position = [39 120 106 15];

app.Label.Text = 'Duljina stranice (b):';

% Create size_b

app.size_b = uieditfield(app.UIFigure, 'numeric');

app.size_b.Position = [192 116 100 22];

% Create LabelNumericEditField2

app.LabelNumericEditField2 = uilabel(app.UIFigure);

app.LabelNumericEditField2.HorizontalAlignment = 'right';

app.LabelNumericEditField2.Position = [39 77 108 15];

app.LabelNumericEditField2.Text = 'Veličina mesha (m):';

% Create mesh_size

app.mesh_size = uieditfield(app.UIFigure, 'numeric');

app.mesh_size.Position = [192 73 100 22];

% Create LabelEditField

app.LabelEditField = uilabel(app.UIFigure);

app.LabelEditField.HorizontalAlignment = 'right';

app.LabelEditField.Position = [325 163 186 15];

app.LabelEditField.Text = 'Naziv excel datoteke za matricu A:';
```

```
            % Create excel_first

            app.excel_first = uieditfield(app.UIFigure, 'text');

            app.excel_first.Position = [526 159 100 22];

            % Create Label2

            app.Label2 = uilabel(app.UIFigure);

            app.Label2.HorizontalAlignment = 'right';

            app.Label2.Position = [325 77 186 15];

            app.Label2.Text = 'Naziv excel datoteke za matricu B:';

            % Create excel_second

            app.excel_second = uieditfield(app.UIFigure, 'text');

            app.excel_second.Position = [526 73 100 22];

            % Create button_matrica_a

            app.button_matrica_a = uibutton(app.UIFigure, 'push');

            app.button_matrica_a.ButtonPushedFcn = createCallbackFcn(app,
@button_matrica_aButtonPushed);

            app.button_matrica_a.Position = [526 116 100 22];

            app.button_matrica_a.Text = 'Izradi matricu A';

            % Create button_matrica_b

            app.button_matrica_b = uibutton(app.UIFigure, 'push');

            app.button_matrica_b.ButtonPushedFcn = createCallbackFcn(app,
@button_matrica_bButtonPushed);

            app.button_matrica_b.Position = [526 30 100 22];

            app.button_matrica_b.Text = 'Izradi matricu B';
```

```
            % Create Label3

            app.Label3 = uilabel(app.UIFigure);

            app.Label3.FontName = 'Microsoft Tai Le';

            app.Label3.Position = [39 200 592 112];

            app.Label3.Text = {'Program izrađuje excel datoteke spremne za
pokretanje macro-a kojim dobijemo putanje po odabranoj površini'; 'u programu
Catia. Moguće je izraditi dvije matrice, koje svaka radi putanje za horizontalno
ili vertikalno'; 'navođenje, ovisno o tome odabiremo matricu ''Izradi matricu
A'' ili ''Izradi matricu B''.'; ''; 'Program radi na principu da se ispune
potrebni podaci, te se klikne na ''Izradi matricu'' gdje nas program navodi';
'da odaberemo ''.dat'' datoteku koju smo prethodno izradili pomoću Catia-e.';
''; 'Excel datoteka će se izraditit u istoj mapi gdje je odabrana ''.dat''
datoteka.'};

        end

    end

    methods (Access = public)

        % Construct app

        function app = Aplikacija_za_putanje()

            % Create and configure components

            createComponents(app)

            % Register the app with App Designer

            registerApp(app, app.UIFigure)

            % Execute the startup function

            runStartupFcn(app, @startupFcn)

            if nargout == 0

                clear app
```

```
            end

        end

        % Code that executes before app deletion

        function delete(app)

            % Delete UIFigure when app is deleted

            delete(app.UIFigure)

        end

    end

end
```