ALMA MATER STUDIORUM

UNIVERSITÀ DI BOLOGNA

SCUOLA DI INGEGNERIA E ARCHITETTURA
SEDE DI FORLÌ

CORSO DI LAUREA IN

INGEGNERIA AEROSPAZIALE

CLASSE L-9

ELABORATO FINALE DI LAUREA IN
MECCANICA DEL VOLO

(ALLEGATO)

MODELLAZIONE E SIMULAZIONE DI SCENARIO MULTIAGENTE CON VELIVOLI MULTIROTORE NON COOPERATIVI

CANDIDATO
AUGUSTO MAZZEI

RELATORE
FABRIZIO GIULIETTI

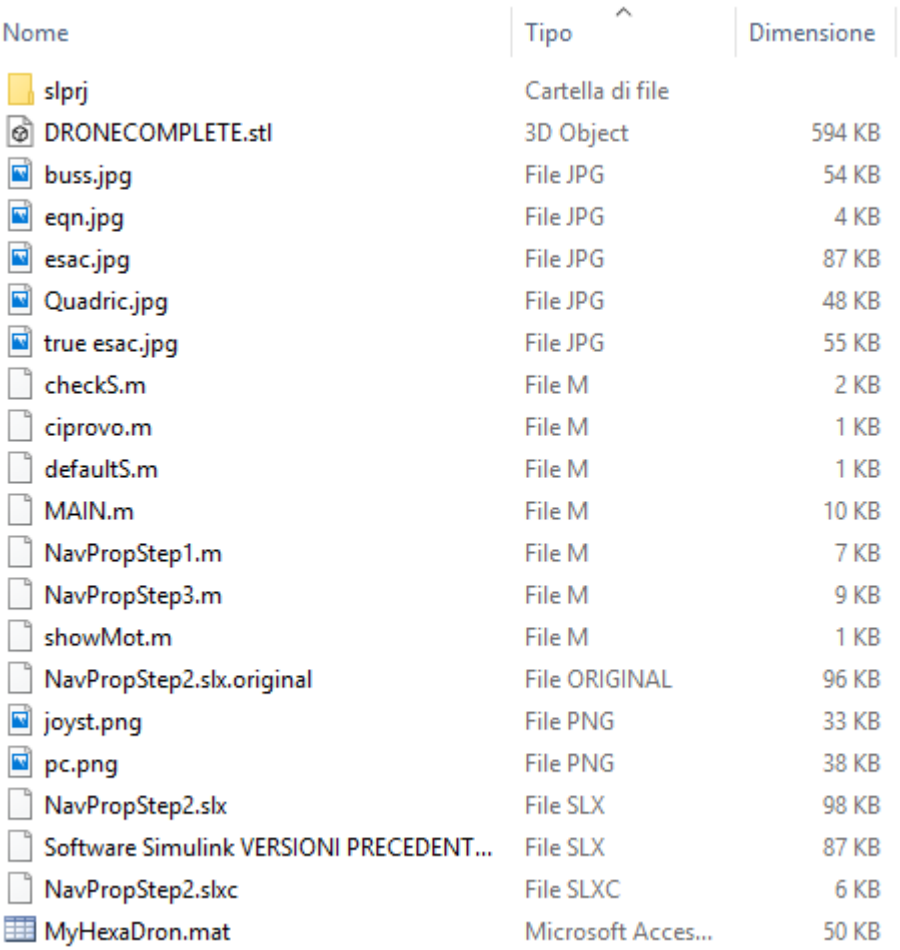
ANNO ACCADEMICO 2017/2018

Sommario

1) ALLEGATO DI TESI – Codice Informatico

- Link per il download dei file in ambiente MATLAB 2018a e Simulink: https://www.dropbox.com/sh/n7bsse6dqmixuh4/AACrnzRRsDSuBJ9evQG-DrMZa?dl=0

- Anteprima del contenuto: (4 MB)



Nome	Tipo	Dimensione
slprj	Cartella di file	
DRONECOMPLETE.stl	3D Object	594 KB
buss.jpg	File JPG	54 KB
eqn.jpg	File JPG	4 KB
esac.jpg	File JPG	87 KB
Quadric.jpg	File JPG	48 KB
true esac.jpg	File JPG	55 KB
checkS.m	File M	2 KB
ciprovo.m	File M	1 KB
defaultS.m	File M	1 KB
MAIN.m	File M	10 KB
NavPropStep1.m	File M	7 KB
NavPropStep3.m	File M	9 KB
showMot.m	File M	1 KB
NavPropStep2.slx.original	File ORIGINAL	96 KB
joyst.png	File PNG	33 KB
pc.png	File PNG	38 KB
NavPropStep2.slx	File SLX	98 KB
Software Simulink VERSIONI PRECEDENT...	File SLX	87 KB
NavPropStep2.slxc	File SLXC	6 KB
MyHexaDron.mat	Microsoft Acces...	50 KB

- Per accedere all'interfaccia eseguire da command window di MATLAB il comando " MAIN "
- Per accedere all'interfaccia Simulink aprire manualmente o con comando "edit" il file NavPropStep2
- È possibile che avvenga l'interruzione del programma in assenza dell'estensione "AEROSPACE BLOCKSET". Un messaggio dalla command window sarà generato automaticamente e l'utente potrà rimediare eseguendo

il download gratuitamente se in presenza di una licenza mathworks (la versione studente è più che sufficiente)

2) Schema Generale del Codice

Fase 1

MAIN.m
apre l'interfaccia principale per l'immissione dei dati ed il lancio della simulazione

defaultS.m
compila automaticamente i dati con valori di default

showMot.m
crea una finestra esterna all'interfaccia in cui visualizzare i coefficienti delle prestazioni del motore adottato nella simulazione

checkS.m
verifica che i dati inseriti non producano errore prima del lancio del software NavPropStep2

Fase 2

NavPropStep2.slx
codice simulink che simula l'inseguimento con le caratteristiche e le conizioni iniziali scelte dall'utente da MAIN.m

Fase 3

NavPropStep3.m
Terminata la simulazione è possibile eseguire questo comando per osservare l'animazione in ambiente MATLAB dell'inseguimento.

Current Folder

DRONECOMPLETE.stl
Geometria del modello 3D del drone in formato STL generata con CATIA. Una volta importata tale geometria è possibile salvarla senza dover riprocedere con la conversione .STL→.mat

MyHexaDron.m
file contenente i vertici e le facce (MyDronF.m & MyDronV.m) del modello 3D del drone S800 EVO

3) MAIN.m

```
close all, clear all
M=figure('Position',[400 100 600 500],'ToolBar','None','Name','MAIN',...
         'Resize','off','NumberTitle','off');
ob={'hunter', 'target' }; phis={'Pos', 'Vel'}; dim={'x','y','z'};
uni={'[m]','[m/s]'};


%in "models" l'utente deve aggiungere il nome di tutti i file .m contenenti
%le coordinate body dei droni che vuole visualizzare nella simulazione.
%le coordinate sono organizzate in 'Faces' e 'Vertices' per l'immissione
%nella funzione 'Patch' presente nella libreria di matlab

% models dovrebbe contenere almeno un file per ogni tipologia di drone:
% un esacottero, e un quadricottero
models={'MyHexaDron'}; Data.eul=[0 0 0];

% 1) motore standard 2) motore sportivo
prestazioni{1}.name='Standard Motor';
prestazioni{1}.thrustLin=[0 35.8907 -681.5058]; %grammi
prestazioni{1}.torqueLin=[0 0.007 -0.0978];
prestazioni{1}.thrust=[0 35.8907 -681.5058];
prestazioni{1}.current=[0 0.2674 -7.166];
prestazioni{1}.rpm=[0 77.1 1117.4];
prestazioni{1}.volt=[0 0 24];

% prestazioni{1}.thrust=[0.3115 1.6264 151.7388];
% prestazioni{1}.current=[0.004 -0.1719 3.5157];
% prestazioni{1}.rpm=[0.2 55.4 1644.3];
% prestazioni{1}.volt=[0 0 24];

prestazioni{2}.name='Sport Motor F1000 kv635';
prestazioni{2}.thrustLin=[0 41.854 -550.7854];
prestazioni{2}.torqueLin=[0 0.004 0.0228];
prestazioni{2}.thrust=[0 41.854 -550.7854]; %grammi
prestazioni{2}.current=[0 0.5077 -11.0663]; %Ampere
prestazioni{2}.rpm=[0 118.456 631.0944]; %grammi
prestazioni{2}.volt=[0 -0.0217 24.5821]; %Volt

% prestazioni{2}.thrust=[0.2712 14.001 -31.6226]; %grammi
% prestazioni{2}.current=[0.0059 -0.0994 0.2497]; %Ampere
% prestazioni{2}.rpm=[-0.3362 152.98 -12.5188]; %grammi
% prestazioni{2}.volt=[-0.0001 -0.015 24.4582]; %Volt

%choose NUMBER MOTORS
cnm = uicontrol(M,'Style','popupmenu','Position',[10,475,100,20],...
                'String',{'6 motors','4 motors'},...
                'Callback','Data.nMot=str2num(cnm.String{cnm.Value}(1));');


%choose TYPE MOTOR
ctm = uicontrol(M,'Style','popupmenu','Position',[130,475,100,20],...

'String',{prestazioni{1}.name,prestazioni{2}.name},'Callback','Data.coef=prestaz
ioni{ctm.Value};');
```

```
uicontrol(M,'Style','pushbutton','Position',[240, 470, 100,25],'String',...
          'Motor Performance','Callback','showMot');

%edit HUNTER MASS
ehm = uicontrol(M,'Style','edit','Position',[470,430,40,20],'String','4',...
                'Callback','Data.m=str2num(ehm.String);');
uicontrol(M,'Style','text','Position',[510,430,80,20],'String','[kg] Mass',...
          'HorizontalAlignment','Left');

%insert hunter ARM LENGHT
ehal = uicontrol(M,'Style','edit','Position',[470,400,40,20],...

'String','0.4','Callback','Data.L=str2num(ehal.String);');
uicontrol(M,'Style','text','Position',[510,400,100,20],'String',...
          '[m] Arms Length','HorizontalAlignment','Left');

%edit DRAG COEFFICIENT
uicontrol(M,'Style','text','Position',[350,350,150,20],'String','CD',...
          'HorizontalAlignment','Left');
for i=1:3
edc{i}=uicontrol(M,'Style','edit','Position',[350,360-i*25,40,20],'String',...
                 [dim{i} ' body'],'Callback',...
                 ['Data.CD(' num2str(i) ')=str2num(edc{' num2str(i)
'}.String);']);
end

%edit hunter INERTIA [kg*m^2]
uicontrol(M,'Style','text','Position',[400,350,150,20],'String','INERTIA',...
          'HorizontalAlignment','Left');
uicontrol(M,'Style','text','Position',[400,360-
4*25,150,20],'String','[kg*m^2]',...
          'HorizontalAlignment','Left');
for i=1:3
ehi{i}=uicontrol(M,'Style','edit','Position',[400,360-i*25,40,20],'String',...
                 [dim{i} ' body'],'Callback',...
                 ['Data.I(' num2str(i) ')=str2num(ehi{' num2str(i)
'}.String);']);
end

%edit hunter SURFACE
uicontrol(M,'Style','text','Position',[445,350,150,20],'String','SURFACE',...
          'HorizontalAlignment','Left');
uicontrol(M,'Style','text','Position',[445,360-4*25,150,20],'String','
[m^2]',...
          'HorizontalAlignment','Left');
for i=1:3
ehs{i}=uicontrol(M,'Style','edit','Position',[450,360-i*25,40,20],'String',...
                 [dim{i} ' body'],'Callback',...
                 ['Data.A(' num2str(i) ')=str2num(ehs{' num2str(i)
'}.String);']);
end

%edit HUNT DELAY
uicontrol(M,'Style','text','Position',[500,225,100,15],'String','HUNT DELAY
[s]',...
          'HorizontalAlignment','Left');
ehd=uicontrol(M,'Style','edit','Position',[500,200,60,20],'String',...
                 '20','Callback','Data.delay=str2num(ehd.String);');
```

```
%edit SAFE HEIGHT
uicontrol(M,'Style','text','Position',[500,175,100,15],'String','SAFE HEIGHT h0
[m]',...
            'HorizontalAlignment','Left');
esh=uicontrol(M,'Style','edit','Position',[500,150,60,20],'String',...
                '0','Callback','Data.h0=str2num(esh.String);');

%edit STOP RANGE
uicontrol(M,'Style','text','Position',[500,125,100,15],'String','STOP RANGE
[m]',...
            'HorizontalAlignment','Left');
esn=uicontrol(M,'Style','edit','Position',[500,100,60,20],'String',...
                '0.8','Callback','Data.range=str2num(esn.String);');

%edit SIMULATION DURATION
uicontrol(M,'Style','text','Position',[500,75,100,15],'String','STOP TIME
[s]',...
            'HorizontalAlignment','Left');
esd=uicontrol(M,'Style','edit','Position',[500,50,60,20],'String',...
                '1000','Callback','Data.time=str2num(esd.String);');

%edit hunter ORIENTATION
angName={'roll','pitch','yaw'};
uicontrol(M,'Style','text','Position',[350,460,150,20],'String',...
          'HUNTER PARAMETERS','HorizontalAlignment','Left');
for i=1:3
eho{i}=uicontrol(M,'Style','edit','Position',[350,390+(i-1)*25,40,20],...
                'String','0','Callback',...
                ['Data.eul(' num2str(i) ')=rem(str2num(eho{' num2str(i)
'}.String),360);',...
                'Vetrina{1}.Vertices=update(MyDronV,Data.eul);',...
                'temp=update1(Data.eul);',...
                'Vetrina{3}.UData=temp(1,:);',...
                'Vetrina{3}.VData=temp(2,:);',...
                'Vetrina{3}.WData=temp(3,:);']);


uicontrol(M,'Style','text','Position',[390,390+(i-1)*25,60,20],'String',...
          ['° ' angName{i}],'HorizontalAlignment','Left');
end

words={'NED','body'};

%edit initial conds
%target Pos, target Vel, hunter Pos, hunter Vel,
uicontrol(M,'Style','text','Position',[10+40+100,160,60,20],'String','NED sdr');
for o = 1:2
   uicontrol(M,'Style','text','Position',[10+40+(o-1)*200,160,60,20],...
         'String',upper(ob{o}),'ForegroundColor',[(o==2) 0 (o==1)]);
    for p = 1:2
    uicontrol(M,'Style','text','Position',[10+(p-1)*80+(o-1)*200,140,60,20],...
              'String',[uni{p} ' ' words{1+(o==1 && p==2)}]);
    for d = 1:3
    eic{o,p,d} = uicontrol(M,'Style','edit','Position',...
                    [10+(o-1)*200+80*(p-1),120-(d-1)*25,60,20],'String',...
                    [dim{d} phis{p}],'Callback',...
        ['Data.' ob{o} '.' phis{p},'(' num2str(d) ')= str2num(eic{' num2str(o)
',' num2str(p) ',' num2str(d) '}.String);']);

    end
```

```
    end
end

%choose running dimension slalom
uicontrol(M,'Style','text','Position',[360,140,120,30],...
          'HorizontalAlignment','left','String','Enable Speed Fluctuation');
esf{1} = uicontrol(M,'Style','checkbox','Position',[375,120,30,20],...
                   'String','X','Callback','Data.Flux(1)=esf{1}.Value;');
esf{2} = uicontrol(M,'Style','checkbox','Position',[375,95,30,20],...
                   'String','Y','Callback','Data.Flux(2)=esf{2}.Value;');
esf{3} = uicontrol(M,'Style','checkbox','Position',[375,70,30,20],...
                    'String','Z','Callback','Data.Flux(3)=esf{3}.Value;');

DS=uicontrol(M,'Style','pushbutton','Position',[28, 20, 90, 35],...
          'String','Default Settings','Callback','defaultS');

VS=uicontrol(M,'Style','pushbutton','Position',[28*2+90, 20, 90, 35],...
          'String','Validate Settings','Callback','checkS');

on = 'on'; run='RUNNING...'; launching='Launch Simulation';
LS=uicontrol(M,'Style','pushbutton','Position',[28*3+90*2, 20, 100, 35],...
          'String','Launch Simulation','Callback',...
          ['checkS; if ok==1, LS.String=run; drawnow, try,',...
          'sim("NavPropStep2",time); success=MYTIME(end)<time, catch,
success=0,',...
          'end, WS.Enable=on; end, LS.String=launching;'],'Enable','off');

WS=uicontrol(M,'Style','pushbutton','Position',[28*4+90*3, 20, 100, 35],...
          'String','Watch Simulation','Callback','NavPropStep3','Enable','off');


AXPL=axes('Position',[0.01,0.37,0.65,0.55],'Units','Normalized','Clipping','off'
);
AXPL.XColor=[1 1 1]; AXPL.YColor=[1 1 1]; AXPL.ZColor=[1 1 1];
AXPL.XTickLabel=[];AXPL.YTickLabel=[];AXPL.ZTickLabel=[];

load(models{cnm.Value},'MyDronF','MyDronV')
% una volta caricato, nel workspace compaiono le variabili 'MyDronF' e
% MyDronV

%creo il modellino del drone
Vetrina{1}=patch('Faces',MyDronF,'Vertices',MyDronV,'FaceColor',[0.3 0.3 0.3]);
view(3), axis equal, hold on, axis(0.5*[-1 1 -1 1 -1 1])

a=0.25; b=-0.25; c=-0.25; Ar=0.15;
Vetrina{2}=quiver3([a a a],[b b b],[c c c],[Ar 0 0],[0 -Ar 0], [0 0 -
Ar],'Color','k');
Vetrina{3}=quiver3([a a a],[b b b],[c c c],[Ar 0 0],[0 -Ar 0], [0 0 -
Ar],'Color','c');



% %
update = @(MyDronV,eul) (eul2rotm([-eul(3),-eul(2),eul(1)] *pi/180)...
         *MyDronV')';

update1 = @(eul) (eul2rotm([-eul(3),-eul(2),eul(1)] *pi/180)*[Ar 0 0; 0 -Ar 0; 0
0 -Ar]);

% %
```

4) defaultS.m

```
%sistema NED (Z ed Y sono invertite rispetto al cartesiano)
Data.hunter.Pos=[randi([-500 500],1,2) -randi(50)]';
Data.hunter.Vel=[0 0 0]';
Data.target.Pos=[0 0 0]';
Data.target.Vel=[randi(1900)/100 0 0]';

Data.eul=[0 0 0];
Data.I=[0.161 0.157 0.218]; %[kg*m^2]
Data.A=[0.065 0.15 0.203]; %[m^2]
Data.CD=[10; 10; 4.5]/6;
Data.Flux=[0 1 1];

for i=1:3
eic{1,1,i}.String=Data.hunter.Pos(i);
eic{1,2,i}.String=Data.hunter.Vel(i);
eic{2,1,i}.String=Data.target.Pos(i);
eic{2,2,i}.String=Data.target.Vel(i);

eho{i}.String=Data.eul(i);
ehs{i}.String=Data.A(i);
ehi{i}.String=Data.I(i);
edc{i}.String=Data.CD(i);
esf{i}.Value=Data.Flux(i);

end
eval(eho{1}.Callback)


Data.time=1000; esd.String='1000';
Data.range=0.8; esn.String='0.8';
Data.h0=0; esh.String='0';
Data.delay=20; ehd.String='20';

Data.m=3;  Data.L=0.4; ehal.String='0.4'; ehm.String='3';
Data.coef=prestazioni{1};
Data.nMot=6; cnm.Value=1; ctm.Value=1;

checkS;
```

5) checkS.m

```
L=Data.L;
g=9.81;
af= Data.coef.thrustLin(2); bf= Data.coef.thrustLin(3);
at= Data.coef.torqueLin(2); bt= Data.coef.torqueLin(3);
a=Data.coef.thrust(1);b=Data.coef.thrust(2); c=Data.coef.thrust(3);
a1=Data.coef.current(1);b1=Data.coef.current(2); c1=Data.coef.current(3);
a2=Data.coef.rpm(1); b2=Data.coef.rpm(2); c2=Data.coef.rpm(3);
a3=Data.coef.volt(1); b3=Data.coef.volt(2); c3=Data.coef.volt(3);


nMot=Data.nMot;
if nMot==6
versione=2;
elseif nMot==4
versione=1;
end

mat6=[ 1           1    1            1            1    1          ;
     -L/2        -L   -L/2          L/2          L    L/2        ;
      L*sin(pi/3) 0   -L*sin(pi/3) -L*sin(pi/3)  0    L*sin(pi/3);
      1          -1    1           -1            1   -1         ];

mat4=[ 1             1              1             1         ;
      -L*cosd(30)   -L*cosd(40)     L*cosd(40)    L*cosd(30);
       L*sind(30)   -L*sind(40)    -L*sind(40)    L*sind(30);
       1             1             -1            -1          ];

invMat6=pinv(mat6);
% Schema dei motori (i dispari ruotano in senso antiorario)
%
%        6     1                 4           1
%
%    5            2
%
%       4      3                 3           2

m=Data.m;
J_x=Data.I(1); J_y=Data.I(2); J_z=Data.I(3);
Ax=Data.A(1); Ay=Data.A(2); Az=Data.A(3);

CD=Data.CD; rho_SL=1.225; beta=9600;
Fx=Data.Flux(1); Fy=Data.Flux(2); Fz=Data.Flux(3);

x_0= Data.hunter.Pos(1);
y_0= Data.hunter.Pos(2);
z_0= Data.hunter.Pos(3);

u_0= Data.hunter.Vel(1);
v_0= Data.hunter.Vel(2);
w_0= Data.hunter.Vel(3);

delay=Data.delay; h0=Data.h0;
time=Data.time; range=Data.range;

phi_0=Data.eul(1)/180*pi;
theta_0=Data.eul(2)/180*pi;
psi_0=Data.eul(3)/180*pi;
```

```
p_0=0; q_0=0; r_0=0;

Vzmax=3; %m/s
Rollmax= 15; %deg°
Pitchmax= 15; %deg°

Vt_start = Data.target.Vel.*[1 -1 -1]';
T_start = Data.target.Pos.*[1 -1 -1]';

LS.Enable='on'; ok=1;
```

6) showMot.m

```
try
  close(M1)
catch
end
M1=figure('Position',[100,100,300,150],'NumberTitle','off','MenuBar','none');
gg=ctm.Value; ee=fieldnames(prestazioni{gg});

for i=1:7
ff=getfield(prestazioni{gg},ee{i});
if i==1
  text(1,8-i,[ ff],'FontSize',12);
else
    text(1,8-i,[ee{i} ':  ' num2str(ff,'%.3f')]);
end
end
axis([0 8 0 8]), xticks([]),yticks([])
```

7) NavPropStep3.m

```
% dopo la fine simulazione
%
clc
% creazione della Figure
F=figure('Color',[1 1 1],'NumberTitle','off','Position',[50 50 800 600]);
AX=axes('Position',[0.05 0.25 0.85 0.75],'Clipping','off');
TT=TargetPos; punta=[.2 0 -.2 0; -.15 .35 -.15 0; 0 0 0 0;]';

% piccole informazioni sulla simulazione
[ah,ahah]=min(Distance.Data);
disp('Distanza minima imposta: [m]'), edge=edge(1);
disp(edge)
disp('Distanza minima raggiunta: [m]')
disp(ah)
disp('differenza di Quota raggiunta in quel momento: [m]')
disp(-Z(ahah)-h0-TT(ahah,3))
disp('differenza di quota minima assoluta raggiunta: [m]')
disp(min(abs(-Z-h0-TT(:,3))))



% Target
T=plot3(TT(1,1),TT(1,2),TT(1,3)+h0 ,'pm'); hold on
TaTrip=plot3(TT(1,1),TT(1,2),TT(1,3),'-r','Clipping','off'); %clip off -> mostra
anche fuori dagli assi
Range=plot3(edge*cos(0:.03:2*pi),edge*sin(0:.03:2*pi),zeros(1,length(0:.03:2*pi)
),'-r');

D=patch('Faces',MyDronF,'Vertices',MyDronV,'FaceColor',[.4 .4
.4],'EdgeColor',[.4 .4 .4]);
axis equal
DrTrip=plot3(X(1),-Y(1),-Z(1),'-b','Clipping','off');
dro=plot3(X(1),-Y(1),0,'xk');
VT=quiver3(X(1),-Y(1),-Z(1),0,0,0,'y','Clipping','off');
O=patch('Faces',MyDronF,'Vertices',MyDronV,'FaceColor','r','EdgeColor','r');

axis auto

%%
% creazione della legenda
Ax2=axes('Position',[0.75 0.01 0.2 0.3]);
Ax2.XTickLabel=''; Ax2.YTickLabel='';Ax2.ZTickLabel='';
Ax2.XTick=[]; Ax2.YTick=[]; Ax2.ZTick=[];
Ax2.XColor=[1 1 1]; Ax2.YColor=[1 1 1]; Ax2.ZColor=[1 1 1];

text(0.5, 1.9, 'HUNTER', 'FontSize',10,'Color','b');
text(1.5, 1.9, 'TARGET', 'FontSize',10,'Color','r');
text(0.05, 1.8, 'POS_{[m]}:', 'FontSize',10);
text(0.05, 1.7, 'H_{[m]}:', 'FontSize',10);
text(0.05, 1.6, 'VEL_{[m/s]}:', 'FontSize',10);

text(0.05, 1.5, 'Assetto°:', 'FontSize',10);
text(1.5, 1.2, ['Range_{[m]}: ' num2str(edge,'%.2f')],
'FontSize',10,'HorizontalAlignment','left');
text(1.5, 1.1, ['h0_{[m]}: ' num2str(h0,'%.2f')], 'FontSize',10);
text(0.05, 1.1, ['V_{hunt}/V_{targ}: '], 'FontSize',10);
```

```
Str= {[ '[' num2str(X(1),'%.1f') ' , ' num2str(-Y(1),'%.1f') ']'],...
      [ '[' num2str(TT(1,1),'%.1f') ' , ' num2str(TT(1,2),'%.1f') ']'],...
      num2str(-Z(1),'%.2f'),...
      num2str(TT(1,3),'%.2f'),...
      num2str(norm([u_0,v_0,w_0]),'%.3f'),...
      num2str(norm(Vtar(1,:)),'%.3f'),...
      [num2str(PHI(1)*180/pi,'%.2f') '_{roll}'],...
      [num2str(THETA(1)*180/pi,'%.2f') '_{pitch}'],...
      [num2str(PSI(1)*180/pi,'%.2f') '_{yaw}'],...
      [num2str(norm([u_0,v_0,w_0])/norm(Vtar(1,:)),'%.3f')]};

StrCord=[0.5,1.8; 1.5,1.8; 0.5,1.7; 1.5,1.7; 0.5,1.6; 1.5,1.6; 0.5,1.4; 0.5,1.3;
0.5,1.2; 0.8,1.1;];

for k=1:length(Str)
S{k}=text(StrCord(k,1),StrCord(k,2),Str{k},'FontSize',8);
end

hold on
Ax2.YLim=[1 2];
Ax2.XLim=[0 2];

%%
% nuovo schema: motori, e manette
MOT=MOT(1);
if MOT==6
THROTTLES=THROT6; mat=mat6;
elseif MOT==4
THROTTLES=THROT4; mat=mat4;
end

Ax4 = axes('Position',[0 0.01 0.3 0.3]);
Ax4.Colormap=autumn;
Ax4.XTickLabel=''; Ax4.YTickLabel=''; Ax4.ZTickLabel='';
Ax4.XTick=[];      Ax4.YTick=[];      Ax4.ZTick=[];
Ax4.YLim=[-1.2 1.2]; Ax4.XLim=[-1.2 1.2];
axis equal, Ax4.XColor=[1 1 1]; Ax4.YColor=[1 1 1]; Ax4.ZColor=[1 1 1];
mTHROT=fix(min(min(THROTTLES)))-3; MTHROT=ceil(max(max(THROTTLES)))+3;
MM=MTHROT-mTHROT;
yea2=colorbar('Ticks',0:.1:1,'TickLabels',num2str(linspace(mTHROT,MTHROT,11)','%
.1f'),'Box','off','Position',[0.3 0.05 0.02 0.17]);

Ax3=axes('Position',[0 0.01 0.3 0.3]); %linkaxes([Ax4 Ax3])
circ=0:0.02:2*pi; the=(60-15*(MOT==4))-(360/MOT)*(0:(MOT-1));
for j=1:MOT
motor{j}=fill(cos(circ)*0.3+cosd(the(j)),sin(circ)*0.3+sind(the(j)),'k');
text(cosd(the(j)),sind(the(j)),num2str(j),'HorizontalAlignment','Center');
hold on
end
Ax3.Colormap=winter;
yea=colorbar('Ticks',[],'Box','off','Position',[0.26 0.05 0.02 0.17]);
axis equal
% preparo le colormap
clockwise = winter; counterclock = autumn;
Ax3.XTickLabel=''; Ax3.YTickLabel=''; Ax3.ZTickLabel='';
Ax3.XTick=[];      Ax3.YTick=[];      Ax3.ZTick=[];
Ax3.YLim=[-1.2 1.2]; Ax3.XLim=[-1.2 1.2];
PP=fill3(punta(:,1),punta(:,2),punta(:,3),'y');
title('Throttles % - Speed Orientation')
```

```
% puntatori mobili dei motori
axis manual
memo=[1.6, 2.3; -0.9, 0.5;];
for k=1:MOT
if mat(end,k)==1
ics=memo(1,1)-(k-1)/2*0.09;
else
ics=memo(1,2)-(k/2-1)*0.09;
end
ipsylon=memo(2,1) +(THROTTLES(1,k)-mTHROT)/MM*(memo(2,2)-memo(2,1));
Joker{k,1}=plot(ics,ipsylon,'>k','Clipping','off'); hold on
Joker{k,2}=text(ics,ipsylon+0.15,num2str(k),'HorizontalAlignment','Center');
end

% Inizia la simulazione
axes(AX); i=0; pause(2)

try
while i<length(PHI)
i=i+10;  i=min(i,length(PHI));

%modifico la visuale
visuale=[TT(i,:); X(i) -Y(i) -Z(i)];
visuale=[min(visuale) max(visuale)];
%AX.XLim=X(i)+[-10 10];
%AX.YLim=-Y(i)+[-10 10];
%AX.ZLim=-Z(i)+[-10 10];

AX.XLim=[visuale(1)-2  visuale(4)+2];
AX.YLim=[visuale(2)-2  visuale(5)+2];
AX.ZLim=[visuale(3)-2  visuale(6)+2];

%modifico il titolo riportando l'istante in secondi di simulazione
F.Name=['istante: ' num2str(Distance.Time(i),'%.2f') ' s'];

%modifico il plot dei droni
Drone1rot=eul2rotm([-PSI(i),-THETA(i),PHI(i)]);
Drone2rot=eul2rotm([atan2(Vt_start(2),Vt_start(1)) 10*pi/180 0]);
points=Drone1rot*MyDronV';
D.Vertices =  [X(i) + points(1,:)'  -Y(i)+points(2,:)' -Z(i)+points(3,:)'];

points=Drone2rot*MyDronV';
O.Vertices =  [TT(i,1) + points(1,:)'  TT(i,2)+points(2,:)'
TT(i,3)+points(3,:)'];

% modifico le traiettorie aggiungendo il nuovo punto
DrTrip.XData(end+(1:10))=  X(i-9:i);
DrTrip.YData(end+(1:10))= -Y(i-9:i);
DrTrip.ZData(end+(1:10))= -Z(i-9:i);

TaTrip.XData(end+(1:10))= TT(i-9:i,1);
TaTrip.YData(end+(1:10))= TT(i-9:i,2);
TaTrip.ZData(end+(1:10))= TT(i-9:i,3);

% modifico la stella 'target'
T.XData= TT(i,1);
T.YData= TT(i,2);
T.ZData= TT(i,3)+h0;
```

```
%aggiorno la traccia a terra del Drone 1
dro.XData=X(i); dro.YData=-Y(i); dro.ZData=TT(i,3)+h0;

%aggiorno il cerchio del Range
Range.XData=edge*cos(0:.03:2*pi)+TT(i,1);
Range.YData=edge*sin(0:.03:2*pi)+TT(i,2);
Range.ZData=(TT(i,3)+h0)*ones(1,length(0:.03:2*pi));

%aggiorno la velocità del Drone 1
vel=Drone1rot*[U(i); -V(i); -W(i)];
VT.XData=X(i); VT.YData=-Y(i); VT.ZData=-Z(i);
VT.UData=vel(1); VT.VData=vel(2); VT.WData=vel(3);

%aggiorno la legenda
Str= {[ '[' num2str(X(i),'%.1f') ' , ' num2str(-Y(i),'%.1f') ']'],...
      [ '[' num2str(TT(i,1),'%.1f') ' , ' num2str(TT(i,2),'%.1f') ']'],...
      num2str(-Z(i),'%.2f'),...
      num2str(TT(i,3),'%.2f'),...
      num2str(norm([U(i),V(i),W(i)]),'%.3f'),...
      num2str(norm(Vtar(i,:)),'%.3f'),...
      [num2str(PHI(i)*180/pi,'%.2f') '_{roll}'],...
      [num2str(THETA(i)*180/pi,'%.2f') '_{pitch}'],...
      [num2str(rem(PSI(i)*180/pi,360),'%.2f') '_{yaw}'],...
      num2str(norm([U(i),V(i),W(i)])/norm(Vtar(i,:)),'%.3f')};

    for z=1:length(Str)
    S{z}.String=Str{z};
    end


%aggiorno le manette
for j=1:MOT
if mat(end,j)==1
motor{j}.FaceColor = clockwise(round((THROTTLES(i,j)-mTHROT)/MM*64),:);
else
motor{j}.FaceColor = counterclock(round((THROTTLES(i,j)-mTHROT)/MM*64),:);
end
end

%aggiorno l'orientamento della velocità (body)
ang1=asin(-W(i)/norm(vel));
ang2=-atan2(V(i),U(i));
PP.Vertices=(eul2rotm([ang2 0 ang1]) *(punta'))';

%muovo i puntatori
for k=1:MOT
ipsylon=memo(2,1) +(THROTTLES(i,k)-mTHROT)/MM*(memo(2,2)-memo(2,1));
Joker{k,1}.YData=ipsylon;
Joker{k,2}.Position(2)=ipsylon+0.15;
Joker{k,2}.Color=motor{k}.FaceColor;
end



drawnow %<--IMPORTANTE
end

% visualizza la traiettoria completa dall'alto
pause(1)
```

```
axes(AX), view(2), IMM=getframe(AX);
pause(2), view(PSI(end)*180/pi,0), pause(3)
view(2)
visuale=[TT; X -Y -Z];
visuale=[min(visuale) max(visuale)];
AX.XLim=[visuale(1)-100  visuale(4)+100];
AX.YLim=[visuale(2)-100  visuale(5)+100];
AX.ZLim=[visuale(3)-5  visuale(6)+30];

% se il drone ha raggiunto l'obiettivo, allora salvo una istantanea.
if Distance.Data(end)<=edge
figure(F);
Ax5 = axes('Position',[0.40 0.01 0.5 0.5]);
Ax5.XTickLabel=''; Ax5.YTickLabel=''; Ax5.ZTickLabel='';
Ax5.XTick=[];      Ax5.YTick=[];      Ax5.ZTick=[];
axes(Ax5)
imshow(IMM.cdata), title("Traiettorie - Vista dall'alto")
end

axes(Ax2); axes(Ax4); axes(Ax3);

%
catch
%do nothing
end
```

8) Conversione da STL a Mat

```
%% estrazione della geometria del drone da un file .stl
clc
model=createpde;
Obj=importGeometry(model,'DRONECOMPLETE.stl');
Target=pdegplot(model);
hold on
Ombra=pdegplot(model);
stampo=cell(1,length(Target));

%trovo il baricentro
xg=mean(mean(Target(1).XData));
yg=mean(mean(Target(1).YData));
zg=mean(mean(Target(1).ZData));

%se il drone non è solidale agli assi lo aggiusto
alf=4; Mat_Cor=[cosd(alf) sind(alf) 0; -sind(alf) cosd(alf) 0; 0 0 1];
for k=1:length(Target)
    for o=1:size(Target(k).XData)
    % riporto il centro di massa nell'origine (0,0,0)
    qq=[Target(k).XData(o,:)-xg;
        Target(k).YData(o,:)-yg;
        Target(k).ZData(o,:)-zg;]/1000;
        qq=Mat_Cor*qq;  % <-- lo aggiusto in modo solidale agli assi x y z

        Target(k).XData(o,:) =qq(1,:); %rimetto tutto nel plot
        Target(k).YData(o,:) =qq(2,:);
        Target(k).ZData(o,:) =qq(3,:);
    end

for z=1:size(Target(k).XData,1) %salvo tutte le coordinate in un cell
stampo{k}{z}=[Target(k).XData(z,:);Target(k).YData(z,:);Target(k).ZData(z,:);];
end
end
Target(1).FaceColor=[0.3 0.3 0.3]; %coloro il drone?

MyDronF=Target(1).Faces;
MyDronV=Target(1).Vertices;
save('MyHexaDron','MyDronF','MyDronV')
close all  <--Il plot 3D del drone è concluso

% Nel caso io lo abbia già fatto potrei evitare tale procedimento salvando
% 'MyDronF' & 'MyDronV' in un file .m pronto all'utilizzo con un semplice
% load:

load MyHexaDron
```