Geometry input data

Starting from a database, a file $.txt$ with all the propeller/rotor/turbine geometry information, the function provides different variables that can be needed for different analysis.

syntax

All inputs are the function identifier as a text, while outputs are structure arrays that contains all the variables needed by each function.

```
[X]= input_per_la_geometria(txt)
```

Where:

INPUT

txt function identifier

OUTPUT

X structure array

example

A test case is shown below, where it is called in input the identifier $'Opti - Prop'$ to which it corresponds the function that provides the axial and the rotational inductions and the thrust and power coefficient distributions of the optimal propeller. In output the function for geometry data input provides the variables needed in input by the function $Opti_Prop.m$ as shown in figure 1.

```
clc; close all; clear all;
X= Geometry_input_data('Opti_Prop');
```

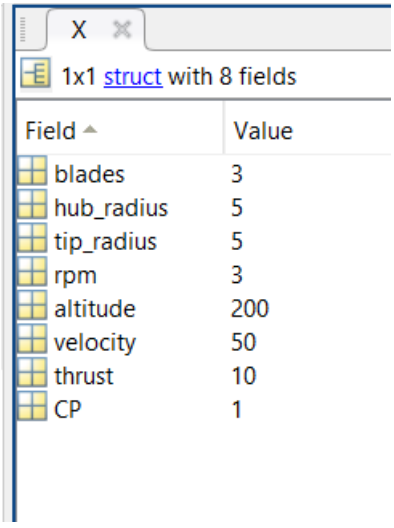

Figure 1: X, structure arrays, output of the function for geometry input data in case of $Opti_Prop.m$

code listing

```
function [X]= Geometry_input_data(txt)

PropDataFileName='propgeometry.txt';    %.txt standard geometry file
myProp=geometryreader(PropDataFileName);       %class call

numdipale = myProp.N ;
raggio= myProp.r ; %adimensional
corda=myProp.c; %adimensional
chords=myProp.chords; %distribution along the radius
radius=myProp.radius; %distribution along the radius
tip_radius= myProp.t_r ;
hub_radius=myProp.h_r ;
thrust=myProp.T;
power=myProp.P;
locknum=myProp.lock_number;
v=myProp.v_inf;
h=myProp.h;
theta=myProp.theta;
pitch=myProp.pitch;
Reynolds=myProp.Reynolds;
Mach=myProp.Mach;
Cd=myProp.Cd;
rpm=myProp.rpm;
f=myProp.f;
advanceratio=myProp.advanceratio;

function_name=txt;
switch function_name
    case 'BEMT'
        X=struct('blades',numdipale, 'r',raggio, 'hub_radius', hub_radius, '
    velocity', v, 'rpm',rpm);
    case 'Darrieus_flusso_multiplo'
        X=struct('corda', corda, 'R', r, 'blades',numdipale,'Cd', Cd);
    case 'Axial_rotor'
        X=struct('r',raggio);
    case 'Ang_attacco_effettivo'
        X=struct('theta',theta);
    case 'elica_intubata'
        X=struct('v',v,'thrust',thrust);
    case 'adim_coeff'
        X=struct('velocity',v,'altitude',h,'blades', numdipale);
    case 'Opti_Prop'
        X=struct('blades',numdipale,'hub_radius', hub_radius,'tip_radius',
    tip_radius,'rpm',rpm,'altitude',h,'velocity', v,'thrust',thrust, 'power'
    , power);
    case 'RvortexInt'
        X=struct('corda',corda);
    case 'flappingangles'
        X=struct('velocity',v, 'Lock number', locknum,'R',r);
    case 'Cdcl_xfoil'
        X=struct('Reynolds number', Reynolds);
    case 'RotorFF'
        X=[]; %they only need in input the angle of attack
    case 'Axial_Descent_Ascent'
        X=struct('R',r);
    case 'Cdcl_xrotor'
        X=struct('Reynolds_number', Reynolds, 'f',f);

end



end
```