

# **Geometrical Construction**

Matlab® Routine (v1.0 – November 2018)

**Giuseppe Scarlatella**

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% Geometrical Construction (v1.0 November 2018) %%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% Created by: Giuseppe Scarlatella
%%% Date: 24/11/2018
%%% Version: 1.0
%%% Edited by: ...
%%% Software based on Matlab® 2017a
%%% Software system based on Windows 10 OS (Apple and Linux compatible)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% Terms of Use: all intellectual rights reserved to Lehrstuhl für
%%% Turbomaschinen und Flugantriebe - Fakultät für Maschinenwesen - TUM.
%%% Any resource relative to the Matlab® Image Analysis Routine must be
%%% shared under specific authorization by Lehrstuhl für Turbomaschinen
%%% und Flugantriebe - Fakultät für Maschinenwesen - TUM and for specific
%%% research purposes only. Any other purpose will violate this Terms of
%%% Use and TUM policies. Any software material, including developer tools
%%% and sample code (collectively "Software"), are subject to this specific
%%% Terms of Use, unless TUM has provided those items to final user under
%%% more specific terms, in which case, those more specific terms will
%%% apply to the relevant item.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% LEFT BLOCK %%%%%%%%%
clc; clear all; close all

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
INPUTS %%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
[to modify] %%%%%%%%%
% TILT ANGLE
% (of 1ST and 2ND Concave Mirrors)
theta = 5;           %[deg]

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
SET UP PROPERTIES %%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
[DO NOT modify] %%%%%%%%%
EFL = 500;           %[mm]
mirror_diam = 50.8;  %[mm]
hub1_slit = 90.4;    %[mm]
hub1_axis1 = 20;     %[mm]
hub2_1STConcMrr = 23.3; %[mm]
hub2_axis1 = 48.9;   %[mm]
root_1STFlatMrr = 18.56; %[mm]
hub2_baseroot = 81.9; %[mm]
root_1STConcMrr = 113; %[mm]
root_1STCncMr_interf = 117; %[mm]

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
ROUTINE %%%%%%%%%
disp(['Mirror Tilt Angle = ', num2str(theta), ' deg']);
disp(['Mirror Focal Lenght = ', num2str(EFL), ' mm']);

theta = deg2rad(theta);

mean_focal_axis = ((hub1_slit*cos(2*theta)-hub1_axis1)+...

```

```

        (hub2_axis1+hub2_1STConcMrr*cos(theta)))/2;
focal_axis_var = abs((hub1_slit*cos(2*theta)-hub1_axis1)-...
        (hub2_axis1+hub2_1STConcMrr*cos(theta)))/2;

B_mean = 2*(EFL/2)*sin(2*theta);
H_mean = (EFL/2)*cos(2*theta);

disp('-----')
X = H_mean + mean_focal_axis + root_1STFlatMrr - 47.5
Y = B_mean + [hub1_axis1+mean_focal_axis]*tan(2*theta) + ...
    [hub2_1STConcMrr*sin(theta)+(mean_focal_axis-...
    (hub2_axis1+hub2_1STConcMrr*cos(theta)))*tan(2*theta)] + ...
    hub2_baseroot - 65

D = Y - root_1STConcMrr;

disp('-----')
disp('Distance (in mm) between')
disp('LightSourceBlock & 1STConcave Mirror is:')
disp(num2str(D))

%%%%%%%%%% VERIFICATION ON theta VALUE %%%%%%%%%%
disp('-----')
% Verification on routine approximation error
% (MAX theta value)
if focal_axis_var >= 5 %[mm]
    if focal_axis_var < 10 %[mm]
        warning(['Routine approx. error is bigger than 5mm. It is mildly',...
            'advisable to adapt smaller tilt angles.']);
        focal_axis_var
    elseif focal_axis_var >= 10 %[mm]
        warning(['Routine approx. error is bigger than 10mm. ',...
            'It is NECESSARY to adapt smaller tilt angles.']);
        focal_axis_var
    end
end
% Verification on light interference error
% (MIN theta value)
% Hp: Area at the mirror supposed to be HALF of 1ST Concave Mirror area
L = (H_mean +...
    (mean_focal_axis-(hub2_axis1+hub2_1STConcMrr*cos(theta)))*...
    tan(2*theta));
MS = L - (3/4)*mirror_diam;
if MS < 0
    warning(['Interference between light beams. ',...
        'It is NECESSARY to adapt bigger tilt angles.']);
    MS
elseif MS <= 5
    warning(['Possible interference between light beams. ',...
        'It is mildly advisable to adapt bigger tilt angles.']);
    MS
end

% Verification on physical interference error

```

```

% (MIN theta value)
interference = Y-root_1STCncMr_interf;
if Y <= root_1STCncMr_interf
    if abs(interference) < 5;
        warning(['Possible physical interference between components ',...
            '(requires verification). It is mildly advisable to adapt ',...
            'bigger tilt angles.']);
        interference
    else
        warning(['Physical interference between components. ',...
            'It is NECESSARY to adapt bigger tilt angles.']);
        interference
    end
end
end

%% RIGHT BLOCK %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
clc; clear all; close all

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% INPUTS %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% [to modify] %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% TILT ANGLE
% (of 1ST and 2ND Concave Mirrors)
theta = 5;                %[deg]

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% SET UP PROPERTIES %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% [DO NOT modify] %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
EFL = 500;                %[mm]
hub2_1STConcMrr = 23.3;   %[mm]
root_1STFlatMrr = 18.56; %[mm]

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% ROUTINE %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
disp(['Mirror Tilt Angle   =   ',num2str(theta),' deg']);
disp(['Mirror Focal Lenght =   ',num2str(EFL),' mm']);

theta = deg2rad(theta);

Z = (EFL/2)*cos(2*theta) + hub2_1STConcMrr*cos(theta) + ...
    root_1STFlatMrr - 47.5

D = EFL*sin(2*theta) + hub2_1STConcMrr*(sin(2*theta)+sin(theta)) - ...
    40 - 25;

disp('-----')
disp('Distance (in mm) between')
disp('2NDConcave Mirror & CameraBlock is:')
disp(num2str(D))

```

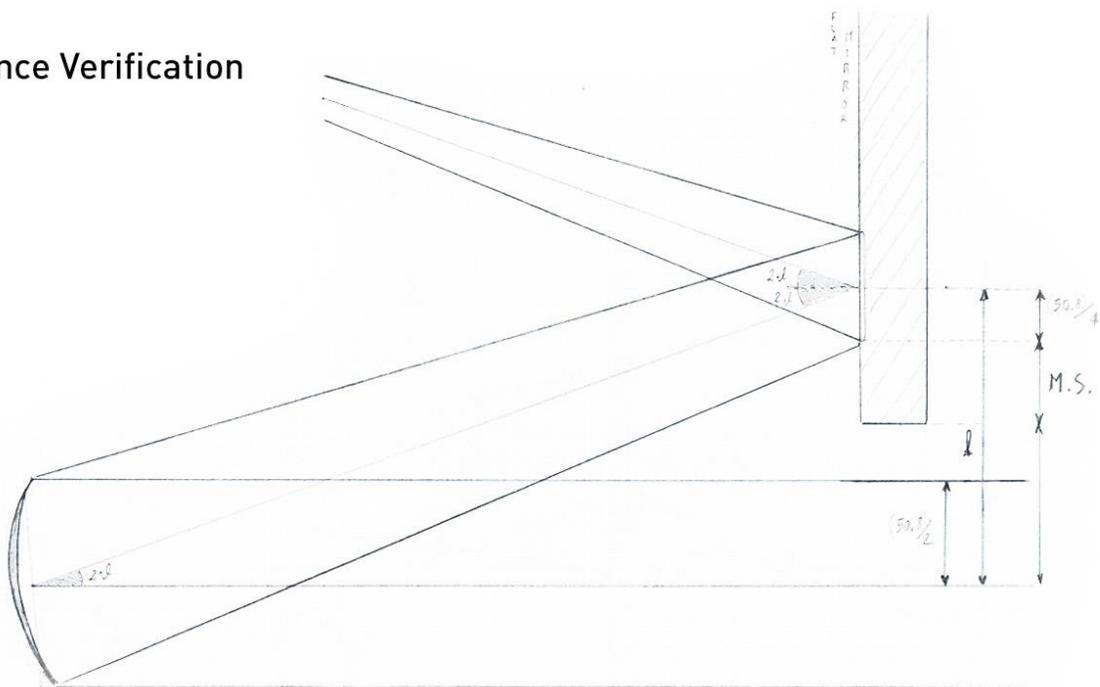
# **Geometrical Construction**

## **Drawings & Schemes**

**Giuseppe Scarlatella**



Interference Verification  
(Sketch)



Consistency Verification  
(Sketch)

