gtrace

Release 0.2.1

Yoichi Aso

Apr 30, 2021

CONTENTS:

CHAPTER

ONE

INTRODUCTION

gtrace is a python package to trace the propagation of Gaussian beams among optical components such as mirrors and lenses. The features of gtrace include:

- Automatically track the Gaussian beam propagation, i.e. q-parameter change, using the ABCD matrix method.
- Reflection and refraction at interface surfaces are properly treated.
- Automatically track the optical distance traveled by a beam through dielectric media.
- Sequential or non-sequential trace modes are available.
- Exporting the results to DXF files

The main motivation behind the development of gtrace was to help the design of the optical layout for the KAGRA interferometer. The task is not so trivial because we had to satisfy many constraints at the same time. Manually placing mirrors on a CAD and adjusting the distances between the mirrors and their orientations was not an option. We needed a way to automatically adjust the optical layout with a computer to satisfy our requirements. In order to do this, we needed a tool to represent an optical layout convenient for computer processings. gtrace represents mirrors and beams as class objects in python. Then the propagation of the beams can be treated as interactions between the beams and the mirrors. This way, we can automate the optimization of the KAGRA optical layout.

The main ingredients of gtrace are mirrors and beams. A mirror is represented as an instance of `Mirror` class. You place mirrors in a 2D plane. You can set various properties of a mirror such as size, curvature, reflectivities, wedge angle etc. Then you launch a beam, an instance of `GaussianBeam` class, from a certain point in the 2D plane. As the beam hits mirrors, it is divided into several sub-beams, such as reflections and deflected beams. These newly generated beam objects are available to you for further propagation. In the non-sequential mode, these sub-beams are automatically propagated until one of the termination conditions is satisfied. The termination conditions include, not hitting any mirror, power is below a certain threshold and so on. At the end of the propagation, you will have a collection of beam objects generated by the collision with the mirrors. You can export the mirrors and the beams into a DXF file.

gtrace is at this moment limited to 2D optical layouts. This limitation might be lifted in the future.

CHAPTER

TWO

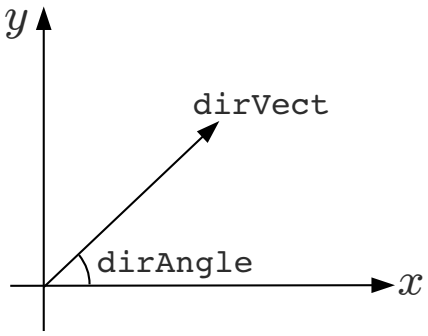
BASIC CONCEPTS

In this section, basic concepts of gtrace will be introduced.

2.1 2D plane

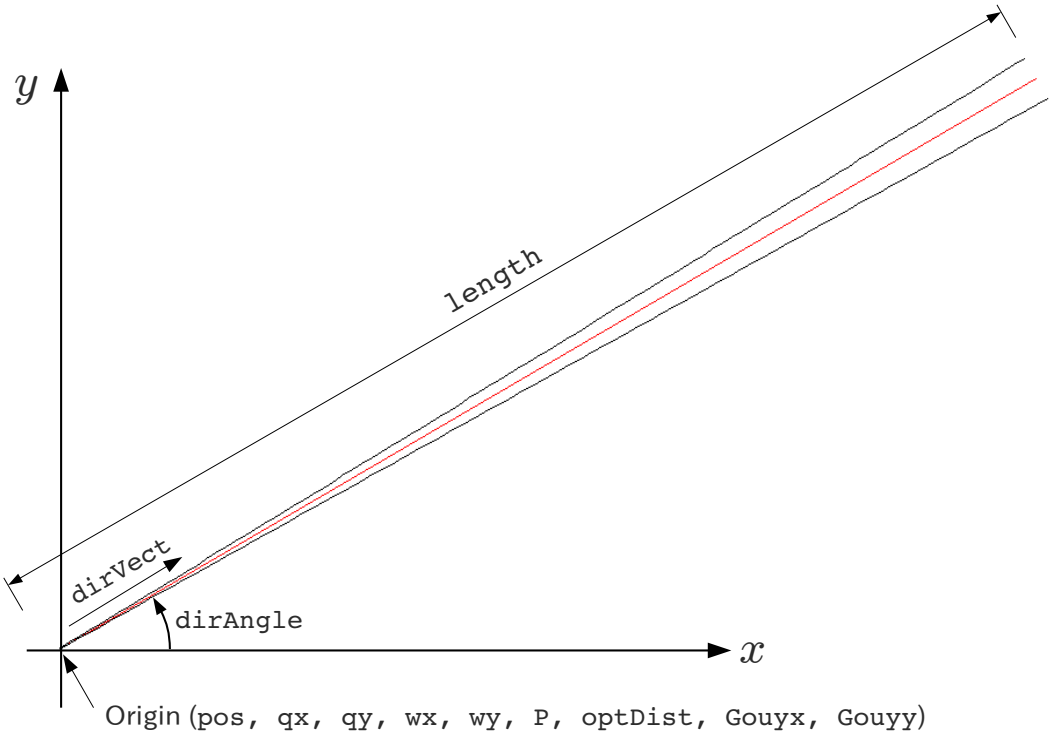
In gtrace world, an optical system will be placed on a two dimentional plane. A location on the plane is specified by a set of Cartesian coordinates (x, y). This just a normal x-y plane. The origin of the axes is at the lower left of the plane. The X-axis extends horizontally to the right. The Y-axis goes up vertically. Nothing more to add here.

2.2 Direction



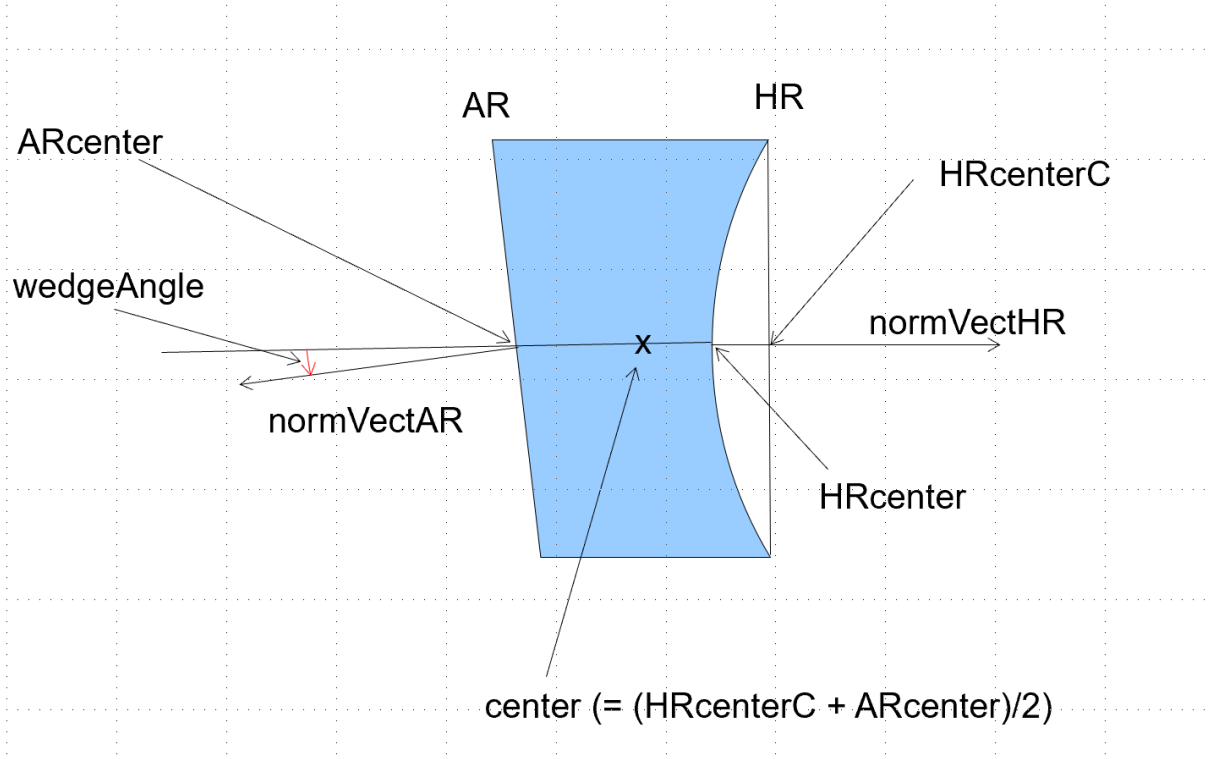
While working with optical layouts, one often has to specify a direction in the 2D plane such as the orientation of a mirror or the propagation direction of a beam. In gtrace, in most cases, a direction can be specified in two ways. One way is to use an angle measured from the X-axis in counter clockwise (`dirAngle` in the figure above). The other way is to use a 2D vector of length 1. If a direction can be specified either way, you only have to specify it in one of those methods. For example, the *GaussianBeam* class has an attribute called `dirVect`. It holds a 2D vector in the form of `numpy.Array`. The *GaussianBeam* class also has an attribute called `dirAngle`, which holds the angle of the beam propagation direction measured from the X-axis in radian. When one of the two attributes is changed, the other is updated automatically to be consistent with the modification. Therefore, you don't have to worry about the consistency. For the direction vector, it is also automatically normalized. Therefore, you can assign it a vector of any norm.

2.3 Beam



A Gaussian beam is represented by an instance of *GaussianBeam* class. The most fundamental properties of a beam is its position (`pos`) and the direction of propagation (`dirVect` or `dirAngle`).

2.4 Mirror



Mirror is a basic optical component in gtrace. Even though the name is "Mirror", it can represent a transparent optical window, a prism, a spherical lense, light absorbing plate (like black glass) and so on. A mirror object has two surfaces, called HR and AR. These surfaces can be flat or curved. Curved surfaces are spherical. If you need a cylindrical surface, use `CyMirror` class instead.

The parameters of a Mirror object are shown in the figure above.

CHAPTER

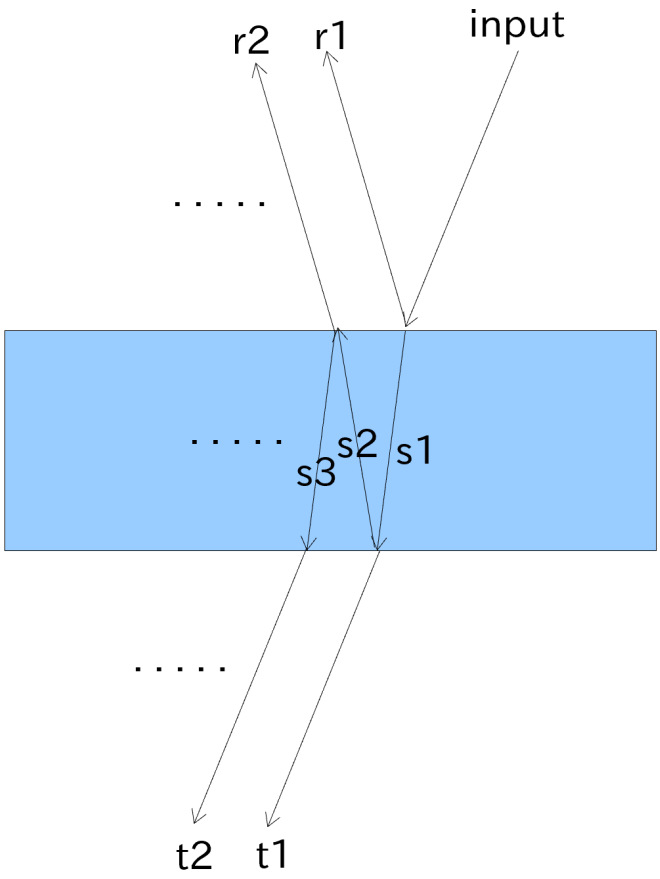
THREE

PROPAGATION OF A BEAM

A beam can be propagated either by manually propagate for a certain distance or tell gtrace to propagate it until hitting a particular mirror.

The manual propagation can be performed by calling, `beam.propagate(d)` where `d` is the distance to propagate.

3.1 Hitting a mirror



By calling `Mirror.hitFromHR(beam)`, you can tell gtrace to propagate the beam until it hits the mirror. If the beam indeed hits the mirror, gtrace will generate a set of beam objects produced by the interactions (reflection and refraction) of the incident beam with the mirror.

The generated beams are given the names indicated in the figure above. The beam objects will be returned as a dictionary with the name of a beam as a key.

CHAPTER

FOUR

TUTORIAL

You can find a basic tutorial of gtrace in its GitHub repository.

You can download the gtrace source tree by:

```
git clone https://github.com/asoy01/gtrace.git
```

Or the direct link to the tutorial file is here: https://github.com/asoy01/gtrace/blob/master/Manuals/Tutorial/gtrace-tutorial.ipynb

Or you can find it directly here:

4.1 gtrace Tutorial

This Jupyter notebook walks you through how to use the gtrace package.

4.1.1 Import modules

First, we import related modules

[1]:
```
import gtrace.beam as beam  # Gaussian beam module
import gtrace.optcomp as opt  # Optical components
import gtrace.draw as draw  # A module to draw results into CAD files
from gtrace.draw.tools import drawAllBeams, drawAllOptics, transAll, rotateAll
↪#Utility functions for drawing
import gtrace.draw.renderer as renderer
from gtrace.unit import *  # A convenience module to represent various modules
import gtrace.optics.gaussian as gauss # A utility module for Gaussian beams
from gtrace.nonsequential import non_seq_trace #Non-sequential trace
from gtrace.optics.geometric import vector_normalize #A function to normalize a vector

import numpy as np  #Numpy
pi = np.pi  #Just for brevity
```

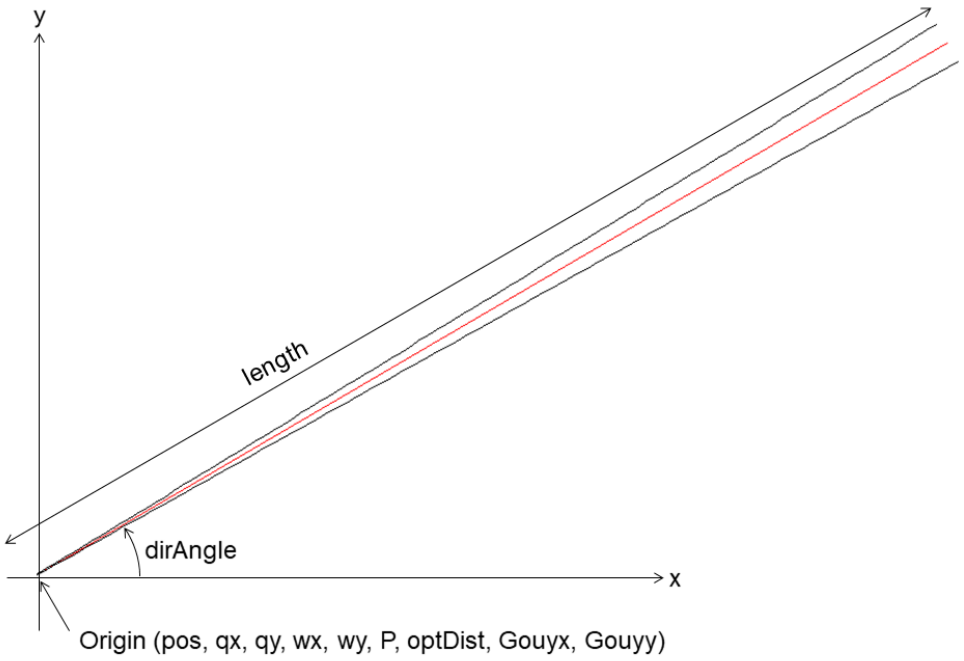
4.1.2 Coordinates and directions

In gtrace, everything is put on a 2D plane and we use a Cartesian coordinate system with x- and y-axis.

Sometimes, you may want to specify a direction in the 2D space, such as a beam propagation direction or the direction of a mirror surface representated by its normal vector. You can use either a unit vector or an angle to specify a direction. When using an angle, it is always measured from the X-axis in counter-clockwise. Therefore, a unit vector having a direction angle θ has components $(\cos\theta, \sin\theta)$.

4.1.3 Gaussian beam object

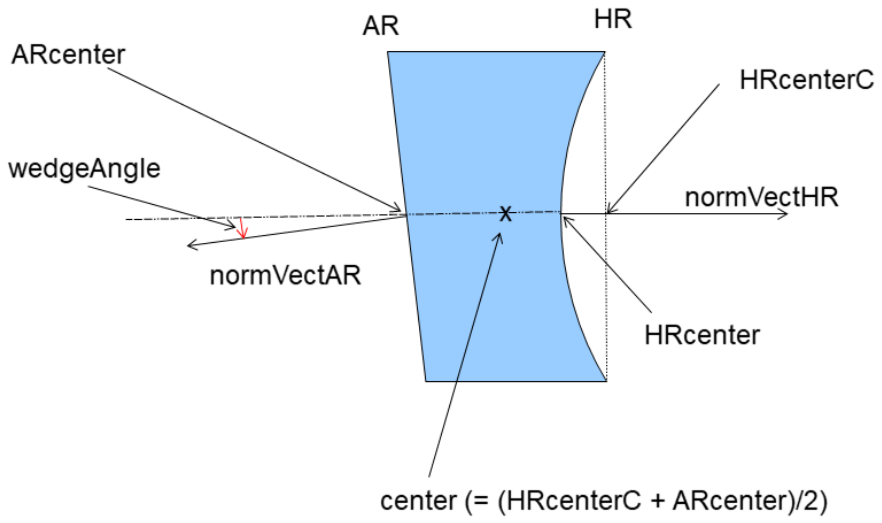
A Gaussian beam object:



[2]:
```
#q-parameter of the beam
q0 = gauss.Rw2q(ROC=np.inf, w=0.3*mm)

#Create a GaussianBeam object.
b0 = beam.GaussianBeam(q0=q0, wl=1064*nm, length=30*cm, P=1.0)

#Set the direction angle of the beam to 10deg from the global x-axis.
b0.dirAngle = deg2rad(10)

#Set the position of the origin of the beam
b0.pos = (0.0, 0.0)
```

4.1.4 Define mirrors

Mirror object:



[5]:
```
M1 = opt.Mirror(HRcenter=[50*cm, 10*cm], normAngleHR=pi,
                diameter=25*cm, thickness=10*cm,
                wedgeAngle=deg2rad(0.25), inv_ROC_HR=1./(120*cm),
                inv_ROC_AR=0,
                Refl_HR=0.9, Trans_HR=1-0.9,
                Refl_AR=500*ppm, Trans_AR=1-500*ppm,
                n=1.45, name='M1')

M2 = opt.Mirror(HRcenter=[0*cm, 18*cm], normAngleHR=deg2rad(5.0),
                diameter=15*cm, thickness=5*cm,
                wedgeAngle=deg2rad(0.25), inv_ROC_HR=-1./(350*cm),
                inv_ROC_AR=0,
                Refl_HR=0.9, Trans_HR=1-0.9,
                Refl_AR=500*ppm, Trans_AR=1-500*ppm,
                n=1.45, name='M2')

M3 = opt.Mirror(HRcenter=[30*cm, 30*cm], normAngleHR=deg2rad(21.3),
                diameter=15*cm, thickness=5*cm,
                wedgeAngle=deg2rad(1), inv_ROC_HR=1./(350*cm),
                inv_ROC_AR=0,
                Refl_HR=0.9, Trans_HR=1-0.9,
                Refl_AR=500*ppm, Trans_AR=1-500*ppm,
                n=1.45, name='M3')
```

4.1.5 Sequential beam trace

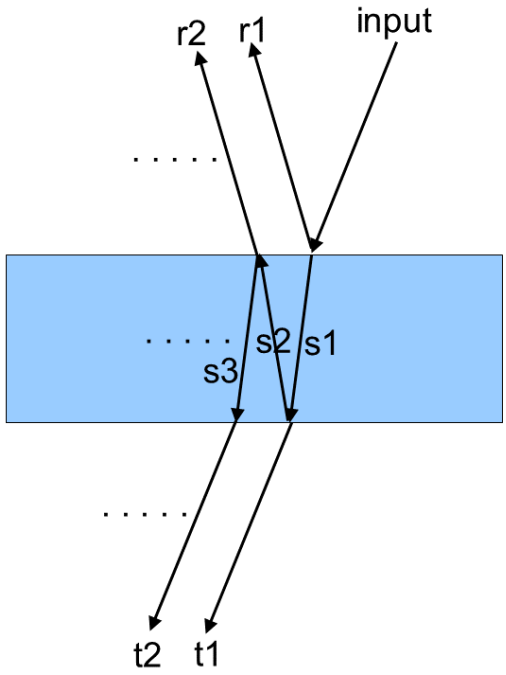
Prepare a dictionary to save beams

[6]:
```
beamDict = {}
```

Hit the mirror `M1` with the beam `b1` from the HR side

[7]:
```
beams = M1.hitFromHR(b0, order=2)
```

Returned object (`beams`) is a dictionary containing the resulting beams. The names of the beams are the following.



Trace the remaining beam path

[9]:
```
#Save the incident beam to the beam dictionary
beamDict['input_beam'] = beams['input']

#Save other beams
beamDict['M1s1'] = beams['s1']
beamDict['M1t1'] = beams['t1']
beamDict['M1s2'] = beams['s2']
beamDict['M1s3'] = beams['s3']
beamDict['M1t2'] = beams['t2']

#Reflected beam from M1
b = beams['r1']

#Hit M2
beams = M2.hitFromHR(b, order=2)

#Save the beam from M1 to M2
beamDict['M1toM2'] = beams['input']
```

(continues on next page)

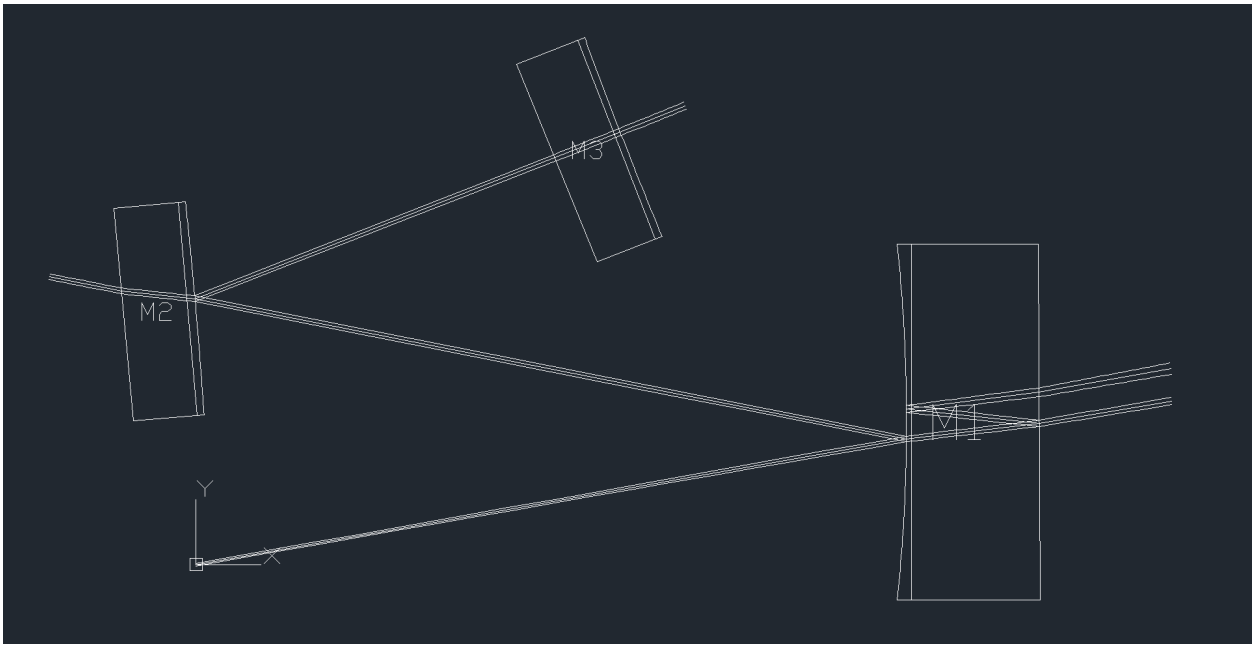
(continued from previous page)

```
beamDict['M2s1'] = beams['s1']
beamDict['M2t1'] = beams['t1']

#Reflected beam from M2
b = beams['r1']

#Hit M3 from AR
beams = M3.hitFromAR(b, order=2)

#Save beams
beamDict['M2toM3'] = beams['input']
beamDict['M3s1'] = beams['s1']
beamDict['M3t1'] = beams['t1']
```

4.1.6 Draw the results

[10]:
```
#Create a canvas object
cnv = draw.Canvas()

#Add a layer to the canvas
cnv.add_layer("main_beam", color=(0,0,0))

#Draw all the beams in beamDict
drawAllBeams(cnv, list(beamDict.values()), drawWidth=True, sigma=3.0, drawPower=False,
                 drawROC=False, drawGouy=False, drawOptDist=False, layer='main_beam',
                    fontSize=0.01)

#Draw the mirror
drawAllOptics(cnv, [M1,M2,M3])

#Save the result as a DXF file
renderer.renderDXF(cnv, 'SeqTrace.dxf')
```

The generated DXF file looks like this.



4.1.7 Non-sequential trace

In the above example, we instructed which beam should hit which optic explicitly. Here, we only specify an input beam and a set of optics. Then let the beam go around until specfied criteria are met. This is called "Non-Sequential Trace".

Perform non-sequential trace

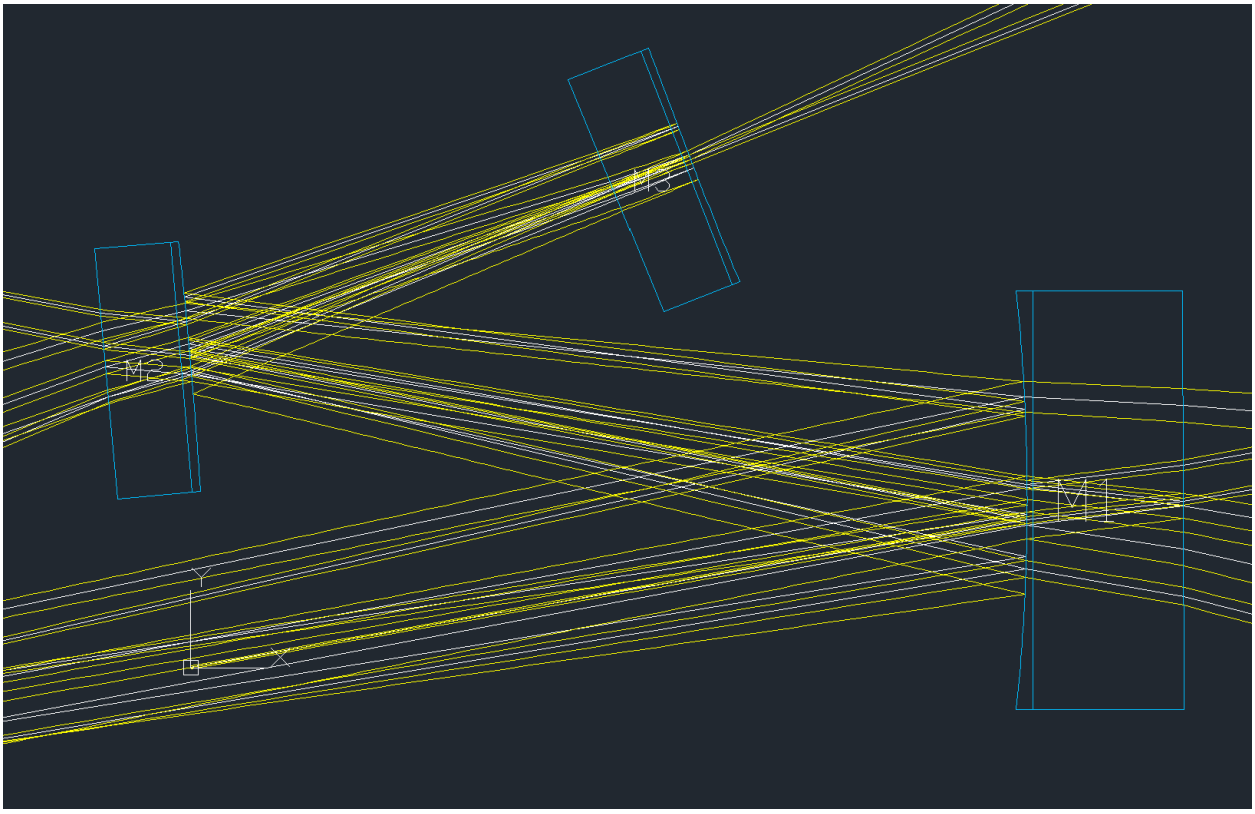
[11]:
```
# Trace beams until the power is less than 1e-6 or 30th order internal reflections
beams = non_seq_trace([M1,M2,M3], b0, order=30, power_threshold=1e-6)
```

Draw results

[13]:
```
#Create a canvas object
cnv = draw.Canvas()

#Add a layer to the canvas
cnv.add_layer("main_beam", color=(0,0,0))

#Draw all beams

drawAllBeams(cnv, beams, drawWidth=True, sigma=3.0, drawPower=False,
                drawROC=False, drawGouy=False, drawOptDist=False, layer='main_beam',
                   fontSize=0.01)

#Draw the mirror
drawAllOptics(cnv, [M1,M2,M3])

#Save the result as a DXF file
renderer.renderDXF(cnv, 'NonSeq.dxf')
```

```
[49]: a=cnv.layers['main_beam']

[52]: b=a.shapes[0]

[54]: b.stop
[54]: (0.4999415205870579, 0.0881531788559148)
```

The generated DXF file looks like this.



4.1.8 KAGRA Input Mode Cleaner

Parameters

```
[14]: nsilica = 1.44967

      MC_Dia = 10.0*cm
      MC_Thick = 3.0*cm
      MCe_ROC = 37.3

      MCi_Refl = 0.9937
      MCo_Refl = 0.9937
      MCe_Refl = 0.9999
      AR_Refl = 0.1/100

      pos_MCi = np.array([-0.25, 0.0])
```

(continues on next page)

(continued from previous page)

```
pos_MCo = np.array([0.25, 0.0])
pos_MCe = np.array([0, 2.63986994e+01])
```

Define MC mirrors

[15]:
```
MCi = opt.Mirror(HRcenter=[0,0], normAngleHR=0.0,
                 diameter=MC_Dia, thickness=MC_Thick,
                 wedgeAngle=-deg2rad(2.5), inv_ROC_HR=0.0,
                 Refl_HR=MCi_Refl, Trans_HR=1-MCi_Refl,
                 Refl_AR=AR_Refl, Trans_AR=1-AR_Refl,
                 n=nsilica, name='MCi')

MCo = opt.Mirror(HRcenter=[0,0], normAngleHR=0.0,
                 diameter=MC_Dia, thickness=MC_Thick,
                 wedgeAngle=deg2rad(2.5), inv_ROC_HR=0.0,
                 Refl_HR=MCo_Refl, Trans_HR=1-MCo_Refl,
                 Refl_AR=AR_Refl, Trans_AR=1-AR_Refl,
                 n=nsilica, name='MCo')

MCe = opt.Mirror(HRcenter=[0,0], normAngleHR=0.0,
                 diameter=MC_Dia, thickness=MC_Thick,
                 wedgeAngle=deg2rad(2.5), inv_ROC_HR=1.0/MCe_ROC,
                 Refl_HR=MCe_Refl, Trans_HR=1-MCe_Refl,
                 Refl_AR=AR_Refl, Trans_AR=1-AR_Refl,
                 n=nsilica, name='MCe')
```

Put MC mirrors

[16]:
```
#Put mirrors in position
MCi.HRcenter = pos_MCi
MCo.HRcenter = pos_MCo
MCe.HRcenter = pos_MCe

#Align the mirrors to form a triangular cavity
v1 = vector_normalize(pos_MCi - pos_MCo)
v2 = vector_normalize(pos_MCe - pos_MCo)
MCo.normVectHR = (v1+v2)/2

v2 = vector_normalize(pos_MCe - pos_MCi)
MCi.normVectHR = (-v1+v2)/2

MCe.normVectHR = np.array([0.0, -1.0])
```

4.1.9 MC eigen mode

Now we compute the eigen mode of the MC by tracing a beam round trip of the MC then extracting the ABCD matrix

[17]:
```
#Test beam (beam parameters does not matter)
b = beam.GaussianBeam(q0=gauss.Rw2q(ROC=np.inf, w=1*mm), wl=1064*nm)

#Put the beam on the surface of MCi
b.pos = MCi.HRcenter

#Direct the beam to the center of MCo
b.dirVect = MCo.HRcenter - b.pos

#Hit MCo
beams = MCo.hitFromHR(b)
b = beams['r1']

#Hit MCe
beams = MCe.hitFromHR(b)
b = beams['r1']

#Hit MCi
beams = MCi.hitFromHR(b)
b = beams['r1']

#Extract the round trip ABCD matrix in the horizontal direction
A = b.Mx[0,0]
B = b.Mx[0,1]
C = b.Mx[1,0]
D = b.Mx[1,1]
#q-parameter of the MC eigenmode beam
qxMC = 1.0/((D-A)/(2*B)-1j*np.sqrt(4-(A+D)**2)/(2*B))

#Extract the round trip ABCD matrix in the vertical direction
A = b.My[0,0]
B = b.My[0,1]
C = b.My[1,0]
D = b.My[1,1]
#q-parameter of the MC eigenmode beam
qyMC = 1.0/((D-A)/(2*B)-1j*np.sqrt(4-(A+D)**2)/(2*B))

#Update the q-parameter of b
b.qx = qxMC
b.qy = qyMC

#Name it bMC
bMC = b.copy()
```

Waist size and position of the MC eigenmode (measured from the MCi HR surface)

[18]:
```
bMC.waist()
```

[18]:
```
{'Waist Size': (0.002388588001545699, 0.002388775565011432),
 'Waist Position': (0.25000000000000416, 0.25000000000000505)}
```

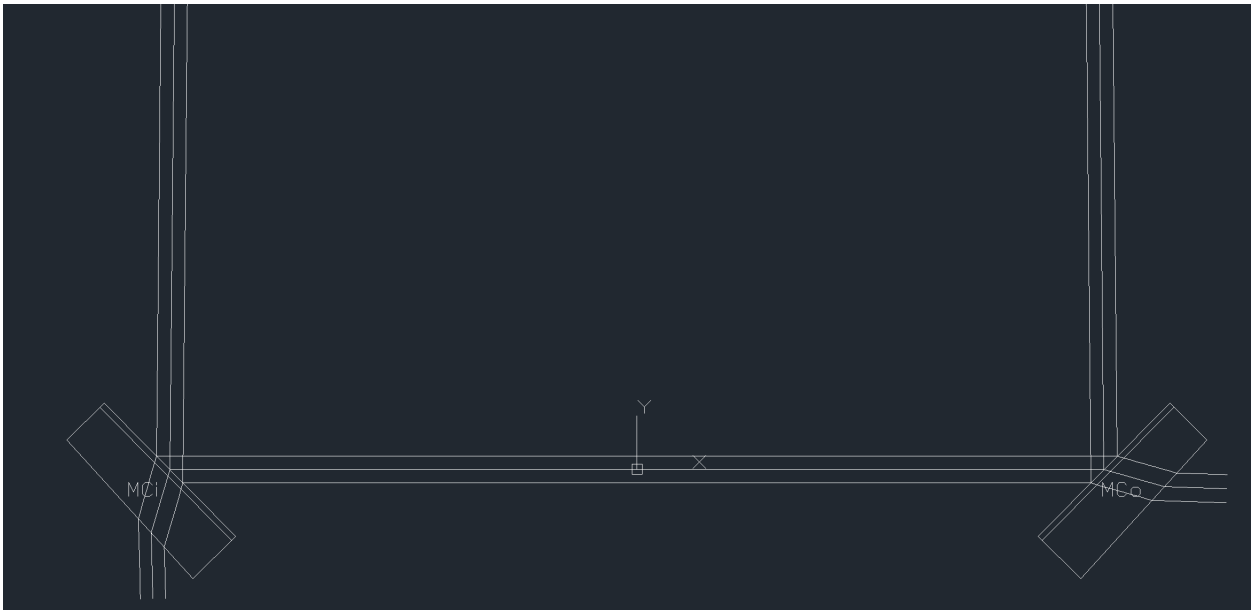
Trace the beam in the MC again

[19]:
```
beamDict = {}

#Hit MCo
beams = MCo.hitFromHR(b, order=1)
beamDict['MCitoMCo'] = beams['input']
beamDict['MCos1'] = beams['s1']
beamDict['MCot1'] = beams['t1']

b = beams['r1']

#Hit MCe
beams = MCe.hitFromHR(b, order=1)
beamDict['MCotoMCe'] = beams['input']
beamDict['MCes1'] = beams['s1']
beamDict['MCet1'] = beams['t1']

b = beams['r1']

#Hit MCi
beams = MCi.hitFromHR(b, order=1)
beamDict['MCetoMCi'] = beams['input']
beamDict['MCis1'] = beams['s1']
beamDict['MCit1'] = beams['t1']
```

4.1.10 Draw MC beams

[20]:
```
#Create a canvas object
cnv = draw.Canvas()

#Add a layer to the canvas
cnv.add_layer("main_beam", color=(0,0,0))

#Draw all beams

drawAllBeams(cnv, list(beamDict.values()), drawWidth=True, sigma=3.0, drawPower=False,
                 drawROC=False, drawGouy=False, drawOptDist=False, layer='main_beam',
                    fontSize=0.01)

#Draw the mirror
drawAllOptics(cnv, [MCi,MCo,MCe])

#Save the result as a DXF file
renderer.renderDXF(cnv, 'MC.dxf')
```

The generated DXF file looks like this:


Y
X
MC1
MC2

[ ]:

CHAPTER

FIVE

API REFERENCE

This page contains auto-generated API reference documentation[1].

5.1 gtrace

This package provides necessary classes and functions for tracing the propagation of Gaussian beams among mirrors and lenses.

5.1.1 Subpackages

gtrace.draw

Submodules

gtrace.draw.draw

Drawing classes for gtrace

Module Contents

Classes

Canvas	Canvas class
Layer	Layer class
Shape	Shape class
Line	Line class
PolyLine	A light weight poly-line
Rectangle	A rectangle
Circle	A circle
Arc	An arc
Text	Text

[1] Created with sphinx-autoapi

Attributes

pi
__author__
__copyright__
__credits__
__license__
__version__
__maintainer__
__email__
__status__

gtrace.draw.draw.**pi**

gtrace.draw.draw.**__author__ = Yoichi Aso**

gtrace.draw.draw.**__copyright__ = Copyright 2011-2021, Yoichi Aso**

gtrace.draw.draw.**__credits__ = ['Yoichi Aso']**

gtrace.draw.draw.**__license__ = BSD**

gtrace.draw.draw.**__version__ = 0.2.1**

gtrace.draw.draw.**__maintainer__ = Yoichi Aso**

gtrace.draw.draw.**__email__ = yoichi.aso@nao.ac.jp**

gtrace.draw.draw.**__status__ = Beta**

class gtrace.draw.draw.**Canvas**(*unit='m'*)
Bases: object

Canvas class

add_layer(*self*, *name*, *color=(0, 0, 0)*)

add_shape(*self*, *shape*, *layername*)

class gtrace.draw.draw.**Layer**(*name*, *color=(0, 0, 0)*)
Bases: object

Layer class

add_shape(*self*, *shape*)

class gtrace.draw.draw.**Shape**
Bases: object

Shape class

class gtrace.draw.draw.**Line**(*start*, *stop*, *thickness=0*)
Bases: *Shape*

Line class

exception gtrace.draw.draw.**NumberOfElementError**
Bases: BaseException

Common base class for all exceptions

__initi__(*self*, *message*)

class gtrace.draw.draw.**PolyLine**(*x*, *y*, *thickness=0*)
Bases: *Shape*

A light weight poly-line

class gtrace.draw.draw.**Rectangle**(*point*, *width*, *height*, *thickness=0*)
Bases: *Shape*

A rectangle

class gtrace.draw.draw.**Circle**(*center*, *radius*, *thickness=0*)
Bases: *Shape*

A circle

class gtrace.draw.draw.**Arc**(*center*, *radius*, *startangle*, *stopangle*, *thickness=0*, *angle_in_rad=True*)
Bases: *Shape*

An arc

Note that angles are stored in rad.

class gtrace.draw.draw.**Text**(*text*, *point*, *height=1.0*, *rotation=0.0*, *angle_in_rad=True*)
Bases: *Shape*

Text

Note that angles are stored in rad.

gtrace.draw.dxf

dxf.py - a DXF export library for python

Sample code:

import dxf d = dxf.DXF('test.dxf') d.add_layer('ABC', color=5) d.add_entity(dxf.Line((1.5,5), (56,-89)), 'ABC') d.save_to_file()

Module Contents

Classes

DXF	A DXF file class.
Layer	Layer class
Entity	A graphic entity
Line	A line entity

continues on next page

Table 3 – continued from previous page

LwPolyLine	A light weight poly-line
Rectangle	A rectangle
Circle	A circle entity
Arc	An arc entity
Text	Text

Functions

color_encode(color)	Given a set of RGB values for a color, find the closest matching
test_func()	

Attributes

pi
color_table
header_template1
header_template2
header_template3
classes_template
tables_template1
vport_template
linetype_template
layer_template1
layer_template2
style_template
view_template
misctable_template
blockrecords_template
tables_template2

continues on next page

Table 5 – continued from previous page

blocks_template
entities_template1
entities_template0
objects_template

gtrace.draw.dxf.**pi**

class gtrace.draw.dxf.**DXF**(*filename='drawing.dxf'*)
Bases: object

A DXF file class.

add_layer(*self*, *name*, *color=1*, *ltype='Continuous'*)

add_entity(*self*, *entity*, *layername*)

save_to_file(*self*)
Save the DXF file

class gtrace.draw.dxf.**Layer**(*name*, *handle*, *color=1*, *ltype='Continuous'*)
Bases: object

Layer class

add_entity(*self*, *entity*)

class gtrace.draw.dxf.**Entity**
Bases: object

A graphic entity

set_handle(*self*, *handle*)

report_min_max(*self*)
Return the coordinates of the lower left and the upper right corners of the drawing. Return value: ((xmin, ymin), (xmax, ymax))

draw(*self*)

class gtrace.draw.dxf.**Line**(*start*, *stop*, *thickness=0*)
Bases: *Entity*

A line entity

report_min_max(*self*)
Return the coordinates of the lower left and the upper right corners of the drawing. Return value: ((xmin, ymin), (xmax, ymax))

draw(*self*, *layername*)

exception gtrace.draw.dxf.**NumberOfElementError**
Bases: BaseException

Common base class for all exceptions

__initi__(*self*, *message*)

class gtrace.draw.dxf.**LwPolyLine**(*x*, *y*, *thickness=0*)
Bases: *Entity*

A light weight poly-line

report_min_max(*self*)
Return the coordinates of the lower left and the upper right corners of the drawing. Return value: ((xmin, ymin), (xmax, ymax))

draw(*self*, *layername*)

class gtrace.draw.dxf.**Rectangle**(*point*, *width*, *height*, *thickness=0*)
Bases: *Entity*

A rectangle

report_min_max(*self*)
Return the coordinates of the lower left and the upper right corners of the drawing. Return value: ((xmin, ymin), (xmax, ymax))

draw(*self*, *layername*)

class gtrace.draw.dxf.**Circle**(*center*, *radius*, *thickness=0*)
Bases: *Entity*

A circle entity

report_min_max(*self*)
Return the coordinates of the lower left and the upper right corners of the drawing. Return value: ((xmin, ymin), (xmax, ymax))

draw(*self*, *layername*)

class gtrace.draw.dxf.**Arc**(*center*, *radius*, *startangle*, *stopangle*, *thickness=0*)
Bases: *Entity*

An arc entity

report_min_max(*self*)
Return the coordinates of the lower left and the upper right corners of the drawing. Return value: ((xmin, ymin), (xmax, ymax))

draw(*self*, *layername*)

class gtrace.draw.dxf.**Text**(*text*, *point*, *height=1.0*, *rotation=0.0*)
Bases: *Entity*

Text

report_min_max(*self*)
Return the coordinates of the lower left and the upper right corners of the drawing. Return value: ((xmin, ymin), (xmax, ymax))

draw(*self*, *layername*)

gtrace.draw.dxf.**color_encode**(*color*)
Given a set of RGB values for a color, find the closest matching one from the pre-defined colors in the DXF specification. Then return its color code (an integer in 1 to 255).

= Input = color: A tuple of three numbers in the range of 0-255, i.e. (R,G,B)

= Return = best_color_num: integer

gtrace.draw.dxf.**color_table**

gtrace.draw.dxf.**header_template1 = Multiline-String**

```
  0
SECTION
  2
HEADER
  9
$ACADVER
  1
AC1024
  9
$ACADMAINTVER
 70
     6
  9
$DWGCODEPAGE
  3
ANSI_1252
  9
$INSBASE
 10
0.0
 20
0.0
 30
0.0
```

gtrace.draw.dxf.**header_template2 = Multiline-String**

```
 9
$EXTMIN
 10
1.0
 20
0.0
 30
0.0
  9
$EXTMAX
 10
73.0
 20
41.0
 30
0.0
  9
$LIMMIN
 10
0.0
 20
0.0
  9
$LIMMAX
 10
1000.0
 20
1000.0
```

gtrace.draw.dxf.**header_template3 = Multiline-String**

```
  9
$ORTHOMODE
 70
     0
  9
$REGENMODE
 70
     1
  9
$FILLMODE
 70
     1
  9
$QTEXTMODE
 70
     0
  9
$MIRRTEXT
 70
     1
  9
$LTSCALE
 40
1.0
  9
$ATTMODE
 70
     1
  9
$TEXTSIZE
 40
2.5
  9
$TRACEWID
 40
1.0
  9
$TEXTSTYLE
  7
Standard
  9
$CLAYER
  8
Outline
  9
$CELTYPE
  6
ByLayer
  9
$CECOLOR
 62
   256
  9
$CELTSCALE
 40
1.0
  9
```

(continues on next page)

(continued from previous page)

```
$DISPSILH
 70
     0
  9
$DIMSCALE
 40
1.0
  9
$DIMASZ
 40
2.5
  9
$DIMEXO
 40
0.625
  9
$DIMDLI
 40
3.75
  9
$DIMRND
 40
0.0
  9
$DIMDLE
 40
0.0
  9
$DIMEXE
 40
1.25
  9
$DIMTP
 40
0.0
  9
$DIMTM
 40
0.0
  9
$DIMTXT
 40
2.5
  9
$DIMCEN
 40
2.5
  9
$DIMTSZ
 40
0.0
  9
$DIMTOL
 70
     0
  9
$DIMLIM
```

(continues on next page)

(continued from previous page)

```
 70
     0
  9
$DIMTIH
 70
     0
  9
$DIMTOH
 70
     0
  9
$DIMSE1
 70
     0
  9
$DIMSE2
 70
     0
  9
$DIMTAD
 70
     1
  9
$DIMZIN
 70
     8
  9
$DIMBLK
  1

  9
$DIMASO
 70
     1
  9
$DIMSHO
 70
     1
  9
$DIMPOST
  1

  9
$DIMAPOST
  1

  9
$DIMALT
 70
     0
  9
$DIMALTD
 70
     3
  9
$DIMALTF
 40
```

(continues on next page)

(continued from previous page)

```
0.03937007874016
  9
$DIMLFAC
 40
1.0
  9
$DIMTOFL
 70
     1
  9
$DIMTVP
 40
0.0
  9
$DIMTIX
 70
     0
  9
$DIMSOXD
 70
     0
  9
$DIMSAH
 70
     0
  9
$DIMBLK1
  1

  9
$DIMBLK2
  1

  9
$DIMSTYLE
  2
ISO-25
  9
$DIMCLRD
 70
     0
  9
$DIMCLRE
 70
     0
  9
$DIMCLRT
 70
     0
  9
$DIMTFAC
 40
1.0
  9
$DIMGAP
 40
0.625
```

(continues on next page)

(continued from previous page)

```
  9
$DIMJUST
 70
     0
  9
$DIMSD1
 70
     0
  9
$DIMSD2
 70
     0
  9
$DIMTOLJ
 70
     0
  9
$DIMTZIN
 70
     8
  9
$DIMALTZ
 70
     0
  9
$DIMALTTZ
 70
     0
  9
$DIMUPT
 70
     0
  9
$DIMDEC
 70
     2
  9
$DIMTDEC
 70
     2
  9
$DIMALTU
 70
     2
  9
$DIMALTTD
 70
     3
  9
$DIMTXSTY
  7
Standard
  9
$DIMAUNIT
 70
     0
  9
```

(continues on next page)

(continued from previous page)

```
$DIMADEC
 70
     0
  9
$DIMALTRND
 40
0.0
  9
$DIMAZIN
 70
     0
  9
$DIMDSEP
 70
    44
  9
$DIMATFIT
 70
     3
  9
$DIMFRAC
 70
     0
  9
$DIMLDRBLK
  1

  9
$DIMLUNIT
 70
     2
  9
$DIMLWD
 70
    -2
  9
$DIMLWE
 70
    -2
  9
$DIMTMOVE
 70
     0
  9
$DIMFXL
 40
1.0
  9
$DIMFXLON
 70
     0
  9
$DIMJOGANG
 40
0.7853981633974483
  9
$DIMTFILL
```

(continues on next page)

(continued from previous page)

```
 70
     0
  9
$DIMTFILLCLR
 70
     0
  9
$DIMARCSYM
 70
     0
  9
$DIMLTYPE
  6

  9
$DIMLTEX1
  6

  9
$DIMLTEX2
  6

  9
$DIMTXTDIRECTION
 70
     0
  9
$LUNITS
 70
     2
  9
$LUPREC
 70
     4
  9
$SKETCHINC
 40
1.0
  9
$FILLETRAD
 40
10.0
  9
$AUNITS
 70
     0
  9
$AUPREC
 70
     0
  9
$MENU
  1
.
  9
$ELEVATION
 40
```

(continues on next page)

(continued from previous page)

```
0.0
  9
$PELEVATION
 40
0.0
  9
$THICKNESS
 40
0.0
  9
$LIMCHECK
 70
     0
  9
$CHAMFERA
 40
10.0
  9
$CHAMFERB
 40
10.0
  9
$CHAMFERC
 40
20.0
  9
$CHAMFERD
 40
0.0
  9
$SKPOLY
 70
     0
  9
$TDCREATE
 40
2456231.748506945
  9
$TDUCREATE
 40
2456231.373506945
  9
$TDUPDATE
 40
2456231.748506945
  9
$TDUUPDATE
 40
2456231.373506945
  9
$TDINDWG
 40
0.0000000116
  9
$TDUSRTIMER
 40
0.0000000116
```

(continues on next page)

(continued from previous page)

```
  9
$USRTIMER
 70
     1
  9
$ANGBASE
 50
0.0
  9
$ANGDIR
 70
     0
  9
$PDMODE
 70
     0
  9
$PDSIZE
 40
0.0
  9
$PLINEWID
 40
0.0
  9
$SPLFRAME
 70
     0
  9
$SPLINETYPE
 70
     6
  9
$SPLINESEGS
 70
     8
  9
$HANDSEED
  5
6F
  9
$SURFTAB1
 70
     6
  9
$SURFTAB2
 70
     6
  9
$SURFTYPE
 70
     6
  9
$SURFU
 70
     6
  9
```

(continues on next page)

(continued from previous page)

```
$SURFV
 70
     6
  9
$UCSBASE
  2

  9
$UCSNAME
  2

  9
$UCSORG
 10
0.0
 20
0.0
 30
0.0
  9
$UCSXDIR
 10
1.0
 20
0.0
 30
0.0
  9
$UCSYDIR
 10
0.0
 20
1.0
 30
0.0
  9
$UCSORTHOREF
  2

  9
$UCSORTHOVIEW
 70
     0
  9
$UCSORGTOP
 10
0.0
 20
0.0
 30
0.0
  9
$UCSORGBOTTOM
 10
0.0
 20
0.0
```

(continues on next page)

(continued from previous page)

```
 30
0.0
  9
$UCSORGLEFT
 10
0.0
 20
0.0
 30
0.0
  9
$UCSORGRIGHT
 10
0.0
 20
0.0
 30
0.0
  9
$UCSORGFRONT
 10
0.0
 20
0.0
 30
0.0
  9
$UCSORGBACK
 10
0.0
 20
0.0
 30
0.0
  9
$PUCSBASE
  2

  9
$PUCSNAME
  2

  9
$PUCSORG
 10
0.0
 20
0.0
 30
0.0
  9
$PUCSXDIR
 10
1.0
 20
0.0
 30
```

(continues on next page)

(continued from previous page)

```
0.0
  9
$PUCSYDIR
 10
0.0
 20
1.0
 30
0.0
  9
$PUCSORTHOREF
  2

  9
$PUCSORTHOVIEW
 70
     0
  9
$PUCSORGTOP
 10
0.0
 20
0.0
 30
0.0
  9
$PUCSORGBOTTOM
 10
0.0
 20
0.0
 30
0.0
  9
$PUCSORGLEFT
 10
0.0
 20
0.0
 30
0.0
  9
$PUCSORGRIGHT
 10
0.0
 20
0.0
 30
0.0
  9
$PUCSORGFRONT
 10
0.0
 20
0.0
 30
0.0
```

(continues on next page)

(continued from previous page)

```
  9
$PUCSORGBACK
 10
0.0
 20
0.0
 30
0.0
  9
$USERI1
 70
     0
  9
$USERI2
 70
     0
  9
$USERI3
 70
     0
  9
$USERI4
 70
     0
  9
$USERI5
 70
     0
  9
$USERR1
 40
0.0
  9
$USERR2
 40
0.0
  9
$USERR3
 40
0.0
  9
$USERR4
 40
0.0
  9
$USERR5
 40
0.0
  9
$WORLDVIEW
 70
     1
  9
$SHADEDGE
 70
     3
  9
```

(continues on next page)

(continued from previous page)

```
$SHADEDIF
 70
    70
  9
$TILEMODE
 70
     1
  9
$MAXACTVP
 70
    64
  9
$PINSBASE
 10
0.0
 20
0.0
 30
0.0
  9
$PLIMCHECK
 70
     0
  9
$PEXTMIN
 10
1.000000000000000E+20
 20
1.000000000000000E+20
 30
1.000000000000000E+20
  9
$PEXTMAX
 10
-1.000000000000000E+20
 20
-1.000000000000000E+20
 30
-1.000000000000000E+20
  9
$PLIMMIN
 10
0.0
 20
0.0
  9
$PLIMMAX
 10
420.0
 20
297.0
  9
$UNITMODE
 70
     0
  9
$VISRETAIN
```

(continues on next page)

(continued from previous page)

```
 70
     1
  9
$PLINEGEN
 70
     0
  9
$PSLTSCALE
 70
     1
  9
$TREEDEPTH
 70
  3020
  9
$CMLSTYLE
  2
Standard
  9
$CMLJUST
 70
     0
  9
$CMLSCALE
 40
20.0
  9
$PROXYGRAPHICS
 70
     1
  9
$MEASUREMENT
 70
     1
  9
$CELWEIGHT
370
    -1
  9
$ENDCAPS
280
     0
  9
$JOINSTYLE
280
     0
  9
$LWDISPLAY
290
     0
  9
$INSUNITS
 70
     4
  9
$HYPERLINKBASE
  1
```

(continues on next page)

(continued from previous page)

```
  9
$STYLESHEET
  1

  9
$XEDIT
290
     1
  9
$CEPSNTYPE
380
     0
  9
$PSTYLEMODE
290
     1
  9
$FINGERPRINTGUID
  2
{92905EEA-4798-D3AF-0738-BBE039DCC74E}
  9
$VERSIONGUID
  2
{FAEB1C32-E019-11D5-929B-00C0DF256EC4}
  9
$EXTNAMES
290
     1
  9
$PSVPSCALE
 40
0.0
  9
$OLESTARTUP
290
     0
  9
$SORTENTS
280
   127
  9
$INDEXCTL
280
     0
  9
$HIDETEXT
280
     1
  9
$XCLIPFRAME
280
     2
  9
$HALOGAP
280
     0
```

(continues on next page)

(continued from previous page)

```
  9
$OBSCOLOR
 70
   257
  9
$OBSLTYPE
280
     0
  9
$INTERSECTIONDISPLAY
280
     0
  9
$INTERSECTIONCOLOR
 70
   257
  9
$DIMASSOC
280
     2
  9
$PROJECTNAME
  1

  9
$CAMERADISPLAY
290
     0
  9
$LENSLENGTH
 40
50.0
  9
$CAMERAHEIGHT
 40
0.0
  9
$STEPSPERSEC
 40
2.0
  9
$STEPSIZE
 40
6.0
  9
$3DDWFPREC
 40
2.0
  9
$PSOLWIDTH
 40
5.0
  9
$PSOLHEIGHT
 40
80.0
  9
```

(continues on next page)

(continued from previous page)

```
$LOFTANG1
 40
1.570796326794897
  9
$LOFTANG2
 40
1.570796326794897
  9
$LOFTMAG1
 40
0.0
  9
$LOFTMAG2
 40
0.0
  9
$LOFTPARAM
 70
     7
  9
$LOFTNORMALS
280
     1
  9
$LATITUDE
 40
37.795
  9
$LONGITUDE
 40
-122.394
  9
$NORTHDIRECTION
 40
0.0
  9
$TIMEZONE
 70
 -8000
  9
$LIGHTGLYPHDISPLAY
280
     1
  9
$TILEMODELIGHTSYNCH
280
     1
  9
$CMATERIAL
347
3C
  9
$SOLIDHIST
280
     1
  9
$SHOWHIST
```

(continues on next page)

(continued from previous page)

```
280
     1
  9
$DWFFRAME
280
     2
  9
$DGNFRAME
280
     2
  9
$REALWORLDSCALE
290
     1
  9
$INTERFERECOLOR
 62
   256
  9
$CSHADOW
280
     0
  9
$SHADOWPLANELOCATION
 40
0.0
  0
ENDSEC
```

gtrace.draw.dxf.**classes_template = Multiline-String**

```
  0
SECTION
  2
CLASSES
  0
CLASS
  1
ACDBDICTIONARYWDFLT
  2
AcDbDictionaryWithDefault
  3
ObjectDBX Classes
 90
        0
 91
        4
280
     0
281
     0
  0
CLASS
  1
VISUALSTYLE
  2
AcDbVisualStyle
```

(continues on next page)

(continued from previous page)

```
  3
ObjectDBX Classes
 90
     4095
 91
         4
280
     0
281
     0
  0
CLASS
  1
MATERIAL
  2
AcDbMaterial
  3
ObjectDBX Classes
 90
     1153
 91
         4
280
     0
281
     0
  0
CLASS
  1
SUN
  2
AcDbSun
  3
SCENEOE
 90
     1024
 91
         4
280
     0
281
     0
  0
ENDSEC
```

gtrace.draw.dxf.**tables_template1 = Multiline-String**

```
  0
SECTION
  2
TABLES
```

gtrace.draw.dxf.**vport_template = Multiline-String**

```
  0
TABLE
  2
```

(continues on next page)

(continued from previous page)

```
VPORT
  5
8
330
0
100
AcDbSymbolTable
 70
     1
  0
VPORT
  5
29
330
8
100
AcDbSymbolTableRecord
100
AcDbViewportTableRecord
  2
*Active
 70
     0
 10
0.0
 20
0.0
 11
1.0
 21
1.0
 12
360.5069142256372
 22
77.37740091485402
 13
0.0
 23
0.0
 14
10.0
 24
10.0
 15
10.0
 25
10.0
 16
0.0
 26
0.0
 36
1.0
 17
0.0
 27
0.0
```

(continues on next page)

(continued from previous page)

```
 37
0.0
 40
643.00113180099
 41
2.116173120728929
 42
50.0
 43
0.0
 44
0.0
 50
0.0
 51
0.0
 71
     0
 72
   100
 73
     1
 74
     3
 75
     0
 76
     0
 77
     0
 78
     0
281
     0
 65
     1
110
0.0
120
0.0
130
0.0
111
1.0
121
0.0
131
0.0
112
0.0
122
1.0
132
0.0
 79
     0
146
```

(continues on next page)

(continued from previous page)

```
0.0
 60
     3
 61
     5
292
     1
282
     1
141
0.0
142
0.0
 63
   250
361
3F
  0
ENDTAB
```

gtrace.draw.dxf.**linetype_template = Multiline-String**

```
  0
TABLE
  2
LTYPE
  5
5
330
0
100
AcDbSymbolTable
 70
     7
  0
LTYPE
  5
14
330
5
100
AcDbSymbolTableRecord
100
AcDbLinetypeTableRecord
  2
ByBlock
 70
     0
  3

 72
    65
 73
     0
 40
0.0
  0
```

(continues on next page)

(continued from previous page)

```
LTYPE
  5
15
330
5
100
AcDbSymbolTableRecord
100
AcDbLinetypeTableRecord
  2
ByLayer
 70
     0
  3

 72
    65
 73
     0
 40
0.0
  0
LTYPE
  5
16
330
5
100
AcDbSymbolTableRecord
100
AcDbLinetypeTableRecord
  2
Continuous
 70
     0
  3
Solid line
 72
    65
 73
     0
 40
0.0
  0
LTYPE
  5
40
330
5
100
AcDbSymbolTableRecord
100
AcDbLinetypeTableRecord
  2
DOT
 70
     0
```

(continues on next page)

(continued from previous page)

```
  3
Dot . . . . . .
 72
    65
 73
     2
 40
2.032
 49
1.016
 74
     0
 49
-1.016
 74
     0
  0
LTYPE
  5
41
330
5
100
AcDbSymbolTableRecord
100
AcDbLinetypeTableRecord
  2
CENTER
 70
     0
  3
Center ____ _ ____ _ ____ _ ____ _ ____ _ ____
 72
    65
 73
     4
 40
13.208
 49
8.128
 74
     0
 49
-1.524
 74
     0
 49
2.032
 74
     0
 49
-1.524
 74
     0
  0
LTYPE
  5
```

(continues on next page)

(continued from previous page)

```
42
330
5
100
AcDbSymbolTableRecord
100
AcDbLinetypeTableRecord
  2
HIDDEN2
 70
     0
  3
Hidden (.5x) _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _
 72
    65
 73
     2
 40
4.762499999999999
 49
3.175
 74
     0
 49
-1.5875
 74
     0
  0
LTYPE
  5
43
330
5
100
AcDbSymbolTableRecord
100
AcDbLinetypeTableRecord
  2
DASHED
 70
     0
  3
Dashed __ __ __ __ __ __ __ __ __ __ __ __ __ _
 72
    65
 73
     2
 40
9.652
 49
8.128
 74
     0
 49
-1.524
 74
     0
```

(continues on next page)

(continued from previous page)

```
  0
LTYPE
  5
44
330
5
100
AcDbSymbolTableRecord
100
AcDbLinetypeTableRecord
  2
PHANTOM2
 70
     0
  3
Phantom (.5x) ___ _ _ ___ _ _ ___ _ _ ___ _ _
 72
    65
 73
     6
 40
14.732
 49
8.128
 74
     0
 49
-1.524
 74
     0
 49
1.016
 74
     0
 49
-1.524
 74
     0
 49
1.016
 74
     0
 49
-1.524
 74
     0
  0
LTYPE
  5
45
330
5
100
AcDbSymbolTableRecord
100
AcDbLinetypeTableRecord
  2
```

(continues on next page)

(continued from previous page)

```
DASHDOT
 70
     0
  3
Dash dot __ . __ . __ . __ . __ . __ . __ . __
 72
    65
 73
     4
 40
11.176
 49
8.128
 74
     0
 49
-1.524
 74
     0
 49
0.0
 74
     0
 49
-1.524
 74
     0
  0
ENDTAB
```

gtrace.draw.dxf.**layer_template1 = Multiline-String**

```
  0
TABLE
  2
LAYER
  5
2
330
0
100
AcDbSymbolTable
 70
     7
  0
LAYER
  5
10
330
2
100
AcDbSymbolTableRecord
100
AcDbLayerTableRecord
  2
0
 70
```

(continues on next page)

(continued from previous page)

```
     0
 62
     5
  6
Continuous
370
    -3
390
F
347
3E
  0
LAYER
  5
46
330
2
100
AcDbSymbolTableRecord
100
AcDbLayerTableRecord
  2
Default
 70
     0
 62
     255
  6
Continuous
370
    -3
390
F
347
3E
```

gtrace.draw.dxf.**layer_template2 = Multiline-String**

```
  0
ENDTAB
```

gtrace.draw.dxf.**style_template = Multiline-String**

```
  0
TABLE
  2
STYLE
  5
3
330
0
100
AcDbSymbolTable
 70
    33
  0
STYLE
```

(continues on next page)

(continued from previous page)

```
  5
11
330
3
100
AcDbSymbolTableRecord
100
AcDbTextStyleTableRecord
  2
Standard
 70
     0
 40
0.0
 41
1.0
 50
0.0
 71
     0
 42
2.5
  3
txt
  4

  0
ENDTAB
```

gtrace.draw.dxf.**view_template = Multiline-String**

```
  0
TABLE
  2
VIEW
  5
6
330
0
100
AcDbSymbolTable
 70
     0
  0
ENDTAB
```

gtrace.draw.dxf.**misctable_template = Multiline-String**

```
  0
TABLE
  2
UCS
  5
7
330
0
100
```

(continues on next page)

(continued from previous page)

```
AcDbSymbolTable
 70
     0
  0
ENDTAB
  0
TABLE
  2
APPID
  5
9
330
0
100
AcDbSymbolTable
 70
     1
  0
APPID
  5
12
330
9
100
AcDbSymbolTableRecord
100
AcDbRegAppTableRecord
  2
ACAD
 70
     0
  0
ENDTAB
  0
TABLE
  2
DIMSTYLE
  5
A
330
0
100
AcDbSymbolTable
 70
     1
100
AcDbDimStyleTable
  0
DIMSTYLE
105
27
330
A
100
AcDbSymbolTableRecord
100
AcDbDimStyleTableRecord
```

(continues on next page)

(continued from previous page)

```
  2
ISO-25
 70
     0
 41
2.5
 42
0.625
 43
3.75
 44
1.25
 73
     0
 74
     0
 77
     1
 78
     8
140
2.5
141
2.5
143
0.03937007874016
147
0.625
171
     3
172
     1
178
     0
271
     2
272
     2
274
     3
278
    44
283
     0
284
     8
340
11
  0
ENDTAB
```

gtrace.draw.dxf.**blockrecords_template = Multiline-String**

```
  0
TABLE
  2
BLOCK_RECORD
```

(continues on next page)

(continued from previous page)

```
  5
1
330
0
100
AcDbSymbolTable
 70
     1
  0
BLOCK_RECORD
  5
1F
330
1
100
AcDbSymbolTableRecord
100
AcDbBlockTableRecord
  2
*Model_Space
340
22
 70
     0
280
     1
281
     0
  0
BLOCK_RECORD
  5
1B
330
1
100
AcDbSymbolTableRecord
100
AcDbBlockTableRecord
  2
*Paper_Space
340
1E
 70
     0
280
     1
281
     0
  0
BLOCK_RECORD
  5
23
330
1
100
AcDbSymbolTableRecord
100
```

(continues on next page)

(continued from previous page)

```
AcDbBlockTableRecord
  2
*Paper_Space0
340
26
 70
     0
280
     1
281
     0
  0
ENDTAB
```

gtrace.draw.dxf.**tables_template2 = Multiline-String**

```
    0
ENDSEC
```

gtrace.draw.dxf.**blocks_template = Multiline-String**

```
  0
SECTION
  2
BLOCKS
  0
BLOCK
  5
20
330
1F
100
AcDbEntity
  8
0
100
AcDbBlockBegin
  2
*Model_Space
 70
     0
 10
0.0
 20
0.0
 30
0.0
  3
*Model_Space
  1

  0
ENDBLK
  5
21
330
1F
```

(continues on next page)

(continued from previous page)

```
100
AcDbEntity
  8
0
100
AcDbBlockEnd
  0
BLOCK
  5
1C
330
1B
100
AcDbEntity
 67
     1
  8
0
100
AcDbBlockBegin
  2
*Paper_Space
 70
     0
 10
0.0
 20
0.0
 30
0.0
  3
*Paper_Space
  1

  0
ENDBLK
  5
1D
330
1B
100
AcDbEntity
 67
     1
  8
0
100
AcDbBlockEnd
  0
BLOCK
  5
24
330
23
100
AcDbEntity
  8
```

(continues on next page)

(continued from previous page)

```
0
100
AcDbBlockBegin
  2
*Paper_Space0
 70
     0
 10
0.0
 20
0.0
 30
0.0
  3
*Paper_Space0
  1

  0
ENDBLK
  5
25
330
23
100
AcDbEntity
  8
0
100
AcDbBlockEnd
  0
ENDSEC
```

gtrace.draw.dxf.**entities_template1 = Multiline-String**

```
  0
SECTION
  2
ENTITIES
```

gtrace.draw.dxf.**entities_template0 = Multiline-String**

```
  0
LINE
  5
7C
330
1F
100
AcDbEntity
  8
Default
100
AcDbLine
 10
1.0
 20
0.0
```

(continues on next page)

(continued from previous page)

```
 30
0.0
 11
73.0
 21
41.0
 31
0.0
  0
ENDSEC
```

gtrace.draw.dxf.**objects_template = Multiline-String**

```
  0
SECTION
  2
OBJECTS
  0
DICTIONARY
  5
C
330
0
100
AcDbDictionary
281
     1
  3
ACAD_GROUP
350
D
  3
ACAD_LAYOUT
350
1A
  3
ACAD_MATERIAL
350
3B
  3
ACAD_MLEADERSTYLE
350
6E
  3
ACAD_MLINESTYLE
350
17
  3
ACAD_PLOTSETTINGS
350
19
  3
ACAD_PLOTSTYLENAME
350
E
  3
ACAD_TABLESTYLE
```

(continues on next page)

(continued from previous page)

```
350
6D
  3
ACAD_VISUALSTYLE
350
2A
  0
SUN
  5
3F
330
29
100
AcDbSun
 90
        1
290
     0
 63
     7
421
 16777215
 40
1.0
291
     1
 91
  2455826
 92
 54000000
292
     0
 70
     2
 71
   256
280
     1
  0
DICTIONARY
  5
D
102
{ACAD_REACTORS
330
C
102
}
330
C
100
AcDbDictionary
281
     1
  0
DICTIONARY
  5
```

(continues on next page)

(continued from previous page)

```
1A
102
{ACAD_REACTORS
330
C
102
}
330
C
100
AcDbDictionary
281
     1
  3
Layout1
350
1E
  3
Layout2
350
26
  3
Model
350
22
  0
DICTIONARY
  5
3B
102
{ACAD_REACTORS
330
C
102
}
330
C
100
AcDbDictionary
281
     1
  3
ByBlock
350
3D
  3
ByLayer
350
3C
  3
Global
350
3E
  0
DICTIONARY
  5
6E
```

(continues on next page)

(continued from previous page)

```
102
{ACAD_REACTORS
330
C
102
}
330
C
100
AcDbDictionary
281
     1
  0
DICTIONARY
  5
17
102
{ACAD_REACTORS
330
C
102
}
330
C
100
AcDbDictionary
281
     1
  3
Standard
350
18
  0
DICTIONARY
  5
19
102
{ACAD_REACTORS
330
C
102
}
330
C
100
AcDbDictionary
281
     1
  0
ACDBDICTIONARYWDFLT
  5
E
102
{ACAD_REACTORS
330
C
102
```

(continues on next page)

(continued from previous page)

```
}
330
C
100
AcDbDictionary
281
     1
  3
Normal
350
F
100
AcDbDictionaryWithDefault
340
F
  0
DICTIONARY
  5
6D
102
{ACAD_REACTORS
330
C
102
}
330
C
100
AcDbDictionary
281
     1
  0
DICTIONARY
  5
2A
102
{ACAD_REACTORS
330
C
102
}
330
C
100
AcDbDictionary
281
     1
  3
2dWireframe
350
2F
  3
3D Hidden
350
31
  3
3dWireframe
```

(continues on next page)

(continued from previous page)

```
350
30
  3
Basic
350
32
  3
Brighten
350
36
  3
ColorChange
350
3A
  3
Conceptual
350
34
  3
Dim
350
35
  3
Facepattern
350
39
  3
Flat
350
2B
  3
FlatWithEdges
350
2C
  3
Gouraud
350
2D
  3
GouraudWithEdges
350
2E
  3
Linepattern
350
38
  3
Realistic
350
33
  3
Thicken
350
37
  0
LAYOUT
  5
```

(continues on next page)

(continued from previous page)

```
1E
102
{ACAD_REACTORS
330
1A
102
}
330
1A
100
AcDbPlotSettings
  1

  2
none_device
  4

  6

 40
0.0
 41
0.0
 42
0.0
 43
0.0
 44
0.0
 45
0.0
 46
0.0
 47
0.0
 48
0.0
 49
0.0
140
0.0
141
0.0
142
1.0
143
1.0
 70
   688
 72
    0
 73
    0
 74
    5
  7

```

(continues on next page)

(continued from previous page)

```
 75
    16
 76
     0
 77
     2
 78
   300
147
1.0
148
0.0
149
0.0
100
AcDbLayout
  1
Layout1
 70
     1
 71
     1
 10
0.0
 20
0.0
 11
420.0
 21
297.0
 12
0.0
 22
0.0
 32
0.0
 14
1.0000000000000000E+20
 24
1.0000000000000000E+20
 34
1.0000000000000000E+20
 15
-1.0000000000000000E+20
 25
-1.0000000000000000E+20
 35
-1.0000000000000000E+20
146
0.0
 13
0.0
 23
0.0
 33
0.0
 16
```

(continues on next page)

(continued from previous page)

```
1.0
 26
0.0
 36
0.0
 17
0.0
 27
1.0
 37
0.0
 76
     0
330
1B
  0
LAYOUT
  5
26
102
{ACAD_REACTORS
330
1A
102
}
330
1A
100
AcDbPlotSettings
  1

  2
none_device
  4

  6

 40
0.0
 41
0.0
 42
0.0
 43
0.0
 44
0.0
 45
0.0
 46
0.0
 47
0.0
 48
0.0
 49
0.0
```

(continues on next page)

(continued from previous page)

```
140
0.0
141
0.0
142
1.0
143
1.0
 70
   688
 72
     0
 73
     0
 74
     5
  7

 75
    16
 76
     0
 77
     2
 78
   300
147
1.0
148
0.0
149
0.0
100
AcDbLayout
  1
Layout2
 70
     1
 71
     2
 10
0.0
 20
0.0
 11
0.0
 21
0.0
 12
0.0
 22
0.0
 32
0.0
 14
0.0
 24
```

(continues on next page)

(continued from previous page)

```
0.0
 34
0.0
 15
0.0
 25
0.0
 35
0.0
146
0.0
 13
0.0
 23
0.0
 33
0.0
 16
1.0
 26
0.0
 36
0.0
 17
0.0
 27
1.0
 37
0.0
 76
     0
330
23
  0
LAYOUT
  5
22
102
{ACAD_REACTORS
330
1A
102
}
330
1A
100
AcDbPlotSettings
  1

  2
none_device
  4

  6

 40
0.0
```

(continues on next page)

(continued from previous page)

```
 41
0.0
 42
0.0
 43
0.0
 44
0.0
 45
0.0
 46
0.0
 47
0.0
 48
0.0
 49
0.0
140
0.0
141
0.0
142
1.0
143
1.0
 70
  1712
 72
     0
 73
     0
 74
     0
  7

 75
     0
 76
     0
 77
     2
 78
   300
147
1.0
148
0.0
149
0.0
100
AcDbLayout
  1
Model
 70
     1
 71
```

(continues on next page)

(continued from previous page)

```
     0
 10
0.0
 20
0.0
 11
420.0
 21
297.0
 12
0.0
 22
0.0
 32
0.0
 14
1.0
 24
0.0
 34
0.0
 15
73.0
 25
41.0
 35
0.0
146
0.0
 13
0.0
 23
0.0
 33
0.0
 16
1.0
 26
0.0
 36
0.0
 17
0.0
 27
1.0
 37
0.0
 76
     0
330
1F
331
29
  0
MATERIAL
  5
3D
```

(continues on next page)

(continued from previous page)

```
102
{ACAD_REACTORS
330
3B
102
}
330
3B
100
AcDbMaterial
  1
ByBlock
 72
     1
 94
      127
  0
MATERIAL
  5
3C
102
{ACAD_REACTORS
330
3B
102
}
330
3B
100
AcDbMaterial
  1
ByLayer
 72
     1
 94
      127
  0
MATERIAL
  5
3E
102
{ACAD_REACTORS
330
3B
102
}
330
3B
100
AcDbMaterial
  1
Global
 72
     1
 94
      127
  0
```

(continues on next page)

(continued from previous page)

```
MLINESTYLE
  5
18
102
{ACAD_REACTORS
330
17
102
}
330
17
100
AcDbMlineStyle
  2
Standard
 70
     0
  3

 62
    256
 51
90.0
 52
90.0
 71
     2
 49
0.5
 62
    256
  6
BYLAYER
 49
-0.5
 62
    256
  6
BYLAYER
  0
ACDBPLACEHOLDER
  5
F
102
{ACAD_REACTORS
330
E
102
}
330
E
  0
VISUALSTYLE
  5
2F
102
{ACAD_REACTORS
```

(continues on next page)

(continued from previous page)

```
330
2A
102
}
330
2A
100
AcDbVisualStyle
  2
2dWireframe
 70
     4
177
     2
291
     0
 71
     0
176
     1
 72
     2
176
     1
 73
     0
176
     1
 90
         0
176
     1
 40
-0.6
176
     1
 41
-30.0
176
     1
 62
     5
 63
     7
421
 16777215
176
     1
 74
     1
176
     1
 91
         4
176
     1
 64
```

(continues on next page)

(continued from previous page)

```
     7
176
     1
 65
   257
176
     1
 75
     1
176
     1
175
     1
176
     1
 42
1.0
176
     1
 92
         0
176
     1
 66
   257
176
     1
 43
1.0
176
     1
 76
     1
176
     1
 77
     6
176
     1
 78
     2
176
     1
 67
     7
176
     1
 79
     5
176
     1
170
     0
176
     1
171
     0
```

(continues on next page)

(continued from previous page)

```
176
     1
290
     0
176
     1
 93
        1
176
     1
 44
0.0
176
     1
173
     0
176
     1
  0
VISUALSTYLE
  5
31
102
{ACAD_REACTORS
330
2A
102
}
330
2A
100
AcDbVisualStyle
  2
3D Hidden
 70
     6
177
     2
291
     0
 71
     1
176
     1
 72
     2
176
     1
 73
     2
176
     1
 90
        0
176
     1
 40
```

(continues on next page)

(continued from previous page)

```
-0.6
176
     1
 41
-30.0
176
     1
 62
     5
 63
     7
421
 16777215
176
     1
 74
     2
176
     1
 91
         2
176
     1
 64
     7
176
     1
 65
   257
176
     1
 75
     2
176
     1
175
     1
176
     1
 42
40.0
176
     1
 92
         0
176
     1
 66
   257
176
     1
 43
1.0
176
     1
 76
     1
```

(continues on next page)

(continued from previous page)

```
176
     1
 77
     6
176
     1
 78
     2
176
     1
 67
     7
176
     1
 79
     3
176
     1
170
     0
176
     1
171
     0
176
     1
290
     0
176
     1
 93
        1
176
     1
 44
0.0
176
     1
173
     0
176
     1
  0
VISUALSTYLE
  5
30
102
{ACAD_REACTORS
330
2A
102
}
330
2A
100
AcDbVisualStyle
  2
```

(continues on next page)

(continued from previous page)

```
3dWireframe
 70
     5
177
     2
291
     0
 71
     0
176
     1
 72
     2
176
     1
 73
     0
176
     1
 90
         0
176
     1
 40
-0.6
176
     1
 41
-30.0
176
     1
 62
     5
 63
     7
421
 16777215
176
     1
 74
     1
176
     1
 91
         4
176
     1
 64
     7
176
     1
 65
   257
176
     1
 75
     1
```

(continues on next page)

(continued from previous page)

```
176
      1
175
      1
176
      1
 42
1.0
176
      1
 92
         0
176
      1
 66
    257
176
      1
 43
1.0
176
      1
 76
      1
176
      1
 77
      6
176
      1
 78
      2
176
      1
 67
      7
176
      1
 79
      5
176
      1
170
      0
176
      1
171
      0
176
      1
290
      0
176
      1
 93
         1
176
```

(continues on next page)

(continued from previous page)

```
     1
 44
0.0
176
     1
173
     0
176
     1
  0
VISUALSTYLE
  5
32
102
{ACAD_REACTORS
330
2A
102
}
330
2A
100
AcDbVisualStyle
  2
Basic
 70
     7
177
     2
291
     1
 71
     1
176
     1
 72
     0
176
     1
 73
     1
176
     1
 90
        0
176
     1
 40
-0.6
176
     1
 41
-30.0
176
     1
 62
     5
```

(continues on next page)

(continued from previous page)

```
 63
     7
421
 16777215
176
     1
 74
     0
176
     1
 91
        4
176
     1
 64
     7
176
     1
 65
   257
176
     1
 75
     1
176
     1
175
     1
176
     1
 42
1.0
176
     1
 92
        8
176
     1
 66
     7
176
     1
 43
1.0
176
     1
 76
     1
176
     1
 77
     6
176
     1
 78
     2
176
```

(continues on next page)

(continued from previous page)

```
     1
 67
     7
176
     1
 79
     5
176
     1
170
     0
176
     1
171
     0
176
     1
290
     0
176
     1
 93
        1
176
     1
 44
0.0
176
     1
173
     0
176
     1
  0
VISUALSTYLE
  5
36
102
{ACAD_REACTORS
330
2A
102
}
330
2A
100
AcDbVisualStyle
  2
Brighten
 70
    12
177
     2
291
     1
 71
     2
```

(continues on next page)

(continued from previous page)

```
176
     1
 72
     2
176
     1
 73
     0
176
     1
 90
        0
176
     1
 40
-0.6
176
     1
 41
-30.0
176
     1
 62
     5
 63
     7
421
 16777215
176
     1
 74
     1
176
     1
 91
        4
176
     1
 64
     7
176
     1
 65
   257
176
     1
 75
     1
176
     1
175
     1
176
     1
 42
1.0
176
```

(continues on next page)

(continued from previous page)

```
     1
 92
        8
176
     1
 66
     7
176
     1
 43
1.0
176
     1
 76
     1
176
     1
 77
     6
176
     1
 78
     2
176
     1
 67
     7
176
     1
 79
     5
176
     1
170
     0
176
     1
171
     0
176
     1
290
     0
176
     1
 93
        1
176
     1
 44
50.0
176
     1
173
     0
176
     1
```

(continues on next page)

(continued from previous page)

```
  0
VISUALSTYLE
  5
3A
102
{ACAD_REACTORS
330
2A
102
}
330
2A
100
AcDbVisualStyle
  2
ColorChange
 70
    16
177
     2
291
     1
 71
     2
176
     1
 72
     2
176
     1
 73
     3
176
     1
 90
        0
176
     1
 40
-0.6
176
     1
 41
-30.0
176
     1
 62
     5
 63
     8
421
  8421504
176
     1
 74
     1
176
```

(continues on next page)

(continued from previous page)

```
     1
 91
        4
176
     1
 64
     7
176
     1
 65
   257
176
     1
 75
     1
176
     1
175
     1
176
     1
 42
1.0
176
     1
 92
        8
176
     1
 66
     8
424
  8421504
176
     1
 43
1.0
176
     1
 76
     1
176
     1
 77
     6
176
     1
 78
     2
176
     1
 67
     7
176
     1
 79
     5
```

(continues on next page)

(continued from previous page)

```
176
     1
170
     0
176
     1
171
     0
176
     1
290
     0
176
     1
 93
        1
176
     1
 44
0.0
176
     1
173
     0
176
     1
  0
VISUALSTYLE
  5
34
102
{ACAD_REACTORS
330
2A
102
}
330
2A
100
AcDbVisualStyle
  2
Conceptual
 70
     9
177
     2
291
     0
 71
     3
176
     1
 72
     2
176
     1
 73
```

(continues on next page)

(continued from previous page)

```
     0
176
     1
 90
         0
176
     1
 40
-0.6
176
     1
 41
-30.0
176
     1
 62
     5
 63
     7
421
 16777215
176
     1
 74
     2
176
     1
 91
         2
176
     1
 64
     7
176
     1
 65
   257
176
     1
 75
     1
176
     1
175
     1
176
     1
 42
40.0
176
     1
 92
         8
176
     1
 66
     7
```

(continues on next page)

(continued from previous page)

```
176
     1
 43
1.0
176
     1
 76
     1
176
     1
 77
     6
176
     1
 78
     2
176
     1
 67
     7
176
     1
 79
     3
176
     1
170
     0
176
     1
171
     0
176
     1
290
     0
176
     1
 93
        1
176
     1
 44
0.0
176
     1
173
     0
176
     1
  0
VISUALSTYLE
  5
35
102
{ACAD_REACTORS
330
```

(continues on next page)

(continued from previous page)

```
2A
102
}
330
2A
100
AcDbVisualStyle
  2
Dim
 70
    11
177
     2
291
     1
 71
     2
176
     1
 72
     2
176
     1
 73
     0
176
     1
 90
         0
176
     1
 40
-0.6
176
     1
 41
-30.0
176
     1
 62
     5
 63
     7
421
 16777215
176
     1
 74
     1
176
     1
 91
         4
176
     1
 64
     7
```

(continues on next page)

(continued from previous page)

```
176
     1
 65
   257
176
     1
 75
     1
176
     1
175
     1
176
     1
 42
1.0
176
     1
 92
         8
176
     1
 66
     7
176
     1
 43
1.0
176
     1
 76
     1
176
     1
 77
     6
176
     1
 78
     2
176
     1
 67
     7
176
     1
 79
     5
176
     1
170
     0
176
     1
171
     0
176
```

(continues on next page)

(continued from previous page)

```
     1
290
     0
176
     1
 93
        1
176
     1
 44
-50.0
176
     1
173
     0
176
     1
  0
VISUALSTYLE
  5
39
102
{ACAD_REACTORS
330
2A
102
}
330
2A
100
AcDbVisualStyle
  2
Facepattern
 70
    15
177
     2
291
     1
 71
     2
176
     1
 72
     2
176
     1
 73
     0
176
     1
 90
        0
176
     1
 40
-0.6
```

(continues on next page)

(continued from previous page)

```
176
     1
 41
-30.0
176
     1
 62
     5
 63
     7
421
 16777215
176
     1
 74
     1
176
     1
 91
        4
176
     1
 64
     7
176
     1
 65
   257
176
     1
 75
     1
176
     1
175
     1
176
     1
 42
1.0
176
     1
 92
        8
176
     1
 66
     7
176
     1
 43
1.0
176
     1
 76
     1
176
```

(continues on next page)

(continued from previous page)

```
     1
 77
     6
176
     1
 78
     2
176
     1
 67
     7
176
     1
 79
     5
176
     1
170
     0
176
     1
171
     0
176
     1
290
     0
176
     1
 93
        1
176
     1
 44
0.0
176
     1
173
     0
176
     1
  0
VISUALSTYLE
  5
2B
102
{ACAD_REACTORS
330
2A
102
}
330
2A
100
AcDbVisualStyle
  2
Flat
```

(continues on next page)

(continued from previous page)

```
 70
     0
177
     2
291
     1
 71
     2
176
     1
 72
     1
176
     1
 73
     1
176
     1
 90
         2
176
     1
 40
-0.6
176
     1
 41
30.0
176
     1
 62
     5
 63
     7
421
 16777215
176
     1
 74
     0
176
     1
 91
         4
176
     1
 64
     7
176
     1
 65
   257
176
     1
 75
     1
176
```

(continues on next page)

(continued from previous page)

```
     1
175
     1
176
     1
 42
1.0
176
     1
 92
        8
176
     1
 66
     7
176
     1
 43
1.0
176
     1
 76
     1
176
     1
 77
     6
176
     1
 78
     2
176
     1
 67
     7
176
     1
 79
     5
176
     1
170
     0
176
     1
171
     0
176
     1
290
     0
176
     1
 93
       13
176
     1
```

(continues on next page)

(continued from previous page)

```
 44
0.0
176
     1
173
     0
176
     1
  0
VISUALSTYLE
  5
2C
102
{ACAD_REACTORS
330
2A
102
}
330
2A
100
AcDbVisualStyle
  2
FlatWithEdges
 70
     1
177
     2
291
     1
 71
     2
176
     1
 72
     1
176
     1
 73
     1
176
     1
 90
        2
176
     1
 40
-0.6
176
     1
 41
30.0
176
     1
 62
     5
 63
```

(continues on next page)

(continued from previous page)

```
     7
421
 16777215
176
     1
 74
     1
176
     1
 91
        4
176
     1
 64
     7
176
     1
 65
   257
176
     1
 75
     1
176
     1
175
     1
176
     1
 42
1.0
176
     1
 92
        0
176
     1
 66
   257
176
     1
 43
1.0
176
     1
 76
     1
176
     1
 77
     6
176
     1
 78
     2
176
     1
```

(continues on next page)

(continued from previous page)

```
 67
     7
176
     1
 79
     5
176
     1
170
     0
176
     1
171
     0
176
     1
290
     0
176
     1
 93
       13
176
     1
 44
0.0
176
     1
173
     0
176
     1
  0
VISUALSTYLE
  5
2D
102
{ACAD_REACTORS
330
2A
102
}
330
2A
100
AcDbVisualStyle
  2
Gouraud
 70
     2
177
     2
291
     1
 71
     2
176
```

(continues on next page)

(continued from previous page)

```
     1
 72
     2
176
     1
 73
     1
176
     1
 90
        2
176
     1
 40
-0.6
176
     1
 41
30.0
176
     1
 62
     5
 63
     7
421
 16777215
176
     1
 74
     0
176
     1
 91
        4
176
     1
 64
     7
176
     1
 65
   257
176
     1
 75
     1
176
     1
175
     1
176
     1
 42
1.0
176
     1
```

(continues on next page)

(continued from previous page)

```
 92
         0
176
      1
 66
      7
176
      1
 43
1.0
176
      1
 76
      1
176
      1
 77
      6
176
      1
 78
      2
176
      1
 67
      7
176
      1
 79
      5
176
      1
170
      0
176
      1
171
      0
176
      1
290
      0
176
      1
 93
        13
176
      1
 44
0.0
176
      1
173
      0
176
      1
  0
```

(continues on next page)

(continued from previous page)

```
VISUALSTYLE
  5
2E
102
{ACAD_REACTORS
330
2A
102
}
330
2A
100
AcDbVisualStyle
  2
GouraudWithEdges
 70
     3
177
     2
291
     1
 71
     2
176
     1
 72
     2
176
     1
 73
     1
176
     1
 90
        2
176
     1
 40
-0.6
176
     1
 41
30.0
176
     1
 62
     5
 63
     7
421
 16777215
176
     1
 74
     1
176
     1
```

(continues on next page)

(continued from previous page)

```
 91
          4
176
      1
 64
      7
176
      1
 65
    257
176
      1
 75
      1
176
      1
175
      1
176
      1
 42
1.0
176
      1
 92
          0
176
      1
 66
    257
176
      1
 43
1.0
176
      1
 76
      1
176
      1
 77
      6
176
      1
 78
      2
176
      1
 67
      7
176
      1
 79
      5
176
      1
170
```

(continues on next page)

(continued from previous page)

```
     0
176
     1
171
     0
176
     1
290
     0
176
     1
 93
       13
176
     1
 44
0.0
176
     1
173
     0
176
     1
  0
VISUALSTYLE
  5
38
102
{ACAD_REACTORS
330
2A
102
}
330
2A
100
AcDbVisualStyle
  2
Linepattern
 70
    14
177
     2
291
     1
 71
     2
176
     1
 72
     2
176
     1
 73
     0
176
     1
```

(continues on next page)

(continued from previous page)

```
 90
         0
176
      1
 40
-0.6
176
      1
 41
-30.0
176
      1
 62
      5
 63
      7
421
 16777215
176
      1
 74
      1
176
      1
 91
         4
176
      1
 64
      7
176
      1
 65
    257
176
      1
 75
      7
176
      1
175
      7
176
      1
 42
1.0
176
      1
 92
         8
176
      1
 66
      7
176
      1
 43
```

(continues on next page)

(continued from previous page)

```
1.0
176
     1
 76
     1
176
     1
 77
     6
176
     1
 78
     2
176
     1
 67
     7
176
     1
 79
     5
176
     1
170
     0
176
     1
171
     0
176
     1
290
     0
176
     1
 93
        1
176
     1
 44
0.0
176
     1
173
     0
176
     1
  0
VISUALSTYLE
  5
33
102
{ACAD_REACTORS
330
2A
102
}
```

(continues on next page)

(continued from previous page)

```
330
2A
100
AcDbVisualStyle
  2
Realistic
 70
     8
177
     2
291
     0
 71
     2
176
     1
 72
     2
176
     1
 73
     0
176
     1
 90
        0
176
     1
 40
-0.6
176
     1
 41
-30.0
176
     1
 62
     5
 63
     7
421
 16777215
176
     1
 74
     1
176
     1
 91
        0
176
     1
 64
     7
176
     1
 65
```

(continues on next page)

(continued from previous page)

```
   257
176
     1
 75
     1
176
     1
175
     1
176
     1
 42
1.0
176
     1
 92
        8
176
     1
 66
     8
424
  7895160
176
     1
 43
1.0
176
     1
 76
     1
176
     1
 77
     6
176
     1
 78
     2
176
     1
 67
     7
176
     1
 79
     5
176
     1
170
     0
176
     1
171
     0
176
     1
```

(continues on next page)

(continued from previous page)

```
290
     0
176
     1
 93
       13
176
     1
 44
0.0
176
     1
173
     0
176
     1
  0
VISUALSTYLE
  5
37
102
{ACAD_REACTORS
330
2A
102
}
330
2A
100
AcDbVisualStyle
  2
Thicken
 70
    13
177
     2
291
     1
 71
     2
176
     1
 72
     2
176
     1
 73
     0
176
     1
 90
        0
176
     1
 40
-0.6
176
```

(continues on next page)

(continued from previous page)

```
     1
 41
-30.0
176
     1
 62
     5
 63
     7
421
 16777215
176
     1
 74
     1
176
     1
 91
        4
176
     1
 64
     7
176
     1
 65
   257
176
     1
 75
     1
176
     1
175
     1
176
     1
 42
1.0
176
     1
 92
       12
176
     1
 66
     7
176
     1
 43
1.0
176
     1
 76
     1
176
     1
```

(continues on next page)

(continued from previous page)

```
 77
     6
176
     1
 78
     2
176
     1
 67
     7
176
     1
 79
     5
176
     1
170
     0
176
     1
171
     0
176
     1
290
     0
176
     1
 93
         1
176
     1
 44
0.0
176
     1
173
     0
176
     1
  0
ENDSEC
```

gtrace.draw.dxf.**test_func**()

gtrace.draw.renderer

Renderer module for gtrace.draw

Module Contents

Functions

renderDXF(canvas, filename)	Render a canvas into a DXF file

Attributes

__author__
__copyright__
__credits__
__license__
__version__
__maintainer__
__email__
__status__

gtrace.draw.renderer.**__author__ = Yoichi Aso**

gtrace.draw.renderer.**__copyright__ = Copyright 2011-2021, Yoichi Aso**

gtrace.draw.renderer.**__credits__ = ['Yoichi Aso']**

gtrace.draw.renderer.**__license__ = BSD**

gtrace.draw.renderer.**__version__ = 0.2.1**

gtrace.draw.renderer.**__maintainer__ = Yoichi Aso**

gtrace.draw.renderer.**__email__ = yoichi.aso@nao.ac.jp**

gtrace.draw.renderer.**__status__ = Beta**

exception gtrace.draw.renderer.**UnknownShapeError**

Bases: BaseException

Common base class for all exceptions

__initi__(*self*, *message*)

gtrace.draw.renderer.**renderDXF**(*canvas*, *filename*)

Render a canvas into a DXF file

gtrace.draw.tools

Module Contents

Functions

drawOptSys(optList, beamList, filename, fontSize=False)	
drawAllBeams(d, beamList, sigma=3.0, drawWidth=True, drawPower=False, drawROC=False, drawGouy=False, drawOptDist=False, layer=None, mode='x', fontSize=0.01)	
drawAllOptics(d, opticsList, drawName=True, layer=None)	
transAll(objList, transVect)	
rotateAll(objList, angle, center)	

Attributes

__author__
__copyright__
__credits__
__license__
__version__
__maintainer__
__email__
__status__

gtrace.draw.tools.**__author__ = Yoichi Aso**

gtrace.draw.tools.**__copyright__ = Copyright 2011-2021, Yoichi Aso**

gtrace.draw.tools.**__credits__ = ['Yoichi Aso']**

gtrace.draw.tools.**__license__ = BSD**

gtrace.draw.tools.**__version__ = 0.2.1**

gtrace.draw.tools.**__maintainer__ = Yoichi Aso**

gtrace.draw.tools.**__email__ = yoichi.aso@nao.ac.jp**

gtrace.draw.tools.**__status__ = Beta**

gtrace.draw.tools.**drawOptSys**(*optList*, *beamList*, *filename*, *fontSize=False*)

gtrace.draw.tools.**drawAllBeams**(*d*, *beamList*, *sigma=3.0*, *drawWidth=True*, *drawPower=False*, *drawROC=False*, *drawGouy=False*, *drawOptDist=False*, *layer=None*, *mode='x'*, *fontSize=0.01*)

gtrace.draw.tools.**drawAllOptics**(*d*, *opticsList*, *drawName=True*, *layer=None*)

gtrace.draw.tools.**transAll**(*objList*, *transVect*)

gtrace.draw.tools.**rotateAll**(*objList*, *angle*, *center*)

Package Contents

Classes

Canvas	Canvas class
Layer	Layer class
Shape	Shape class
Line	Line class
PolyLine	A light weight poly-line
Rectangle	A rectangle
Circle	A circle
Arc	An arc
Text	Text

Attributes

__author__
__copyright__
__credits__
__license__
__version__
__maintainer__
__email__
__status__
pi
__author__
__copyright__

continues on next page

Table 11 – continued from previous page

__credits__
__license__
__version__
__maintainer__
__email__
__status__

gtrace.draw.**__author__ = Yoichi Aso**

gtrace.draw.**__copyright__ = Copyright 2011-2021, Yoichi Aso**

gtrace.draw.**__credits__ = ['Yoichi Aso']**

gtrace.draw.**__license__ = BSD**

gtrace.draw.**__version__ = 0.2.1**

gtrace.draw.**__maintainer__ = Yoichi Aso**

gtrace.draw.**__email__ = yoichi.aso@nao.ac.jp**

gtrace.draw.**__status__ = Beta**

gtrace.draw.**pi**

gtrace.draw.**__author__ = Yoichi Aso**

gtrace.draw.**__copyright__ = Copyright 2011-2021, Yoichi Aso**

gtrace.draw.**__credits__ = ['Yoichi Aso']**

gtrace.draw.**__license__ = BSD**

gtrace.draw.**__version__ = 0.2.1**

gtrace.draw.**__maintainer__ = Yoichi Aso**

gtrace.draw.**__email__ = yoichi.aso@nao.ac.jp**

gtrace.draw.**__status__ = Beta**

class gtrace.draw.**Canvas**(*unit='m'*)

Bases: object

Canvas class

add_layer(*self*, *name*, *color=(0, 0, 0)*)

add_shape(*self*, *shape*, *layername*)

class gtrace.draw.**Layer**(*name*, *color=(0, 0, 0)*)

Bases: object

Layer class

add_shape(*self*, *shape*)

class gtrace.draw.**Shape**
Bases: object

Shape class

class gtrace.draw.**Line**(*start*, *stop*, *thickness=0*)
Bases: *Shape*

Line class

exception gtrace.draw.**NumberOfElementError**
Bases: BaseException

Common base class for all exceptions

__initi__(*self*, *message*)

class gtrace.draw.**PolyLine**(*x*, *y*, *thickness=0*)
Bases: *Shape*

A light weight poly-line

class gtrace.draw.**Rectangle**(*point*, *width*, *height*, *thickness=0*)
Bases: *Shape*

A rectangle

class gtrace.draw.**Circle**(*center*, *radius*, *thickness=0*)
Bases: *Shape*

A circle

class gtrace.draw.**Arc**(*center*, *radius*, *startangle*, *stopangle*, *thickness=0*, *angle_in_rad=True*)
Bases: *Shape*

An arc

Note that angles are stored in rad.

class gtrace.draw.**Text**(*text*, *point*, *height=1.0*, *rotation=0.0*, *angle_in_rad=True*)
Bases: *Shape*

Text

Note that angles are stored in rad.

gtrace.optics

gtrace.optics

This package provides utility functions and classes for Gaussian optics.

Submodules

gtrace.optics.cavity

cavity.py - A Cavity class and related functions for representing a Fabry-Perot cavity

Module Contents

Classes

Cavity	A class to represent a Fabry-Perot cavity.

Functions

finesse(r1, r2, power=False)	Returns the finesse of a cavity

Attributes

pi
c
sqrt

gtrace.optics.cavity.**pi**

gtrace.optics.cavity.**c = 299792458.0**

gtrace.optics.cavity.**sqrt**

class gtrace.optics.cavity.**Cavity**(*r1=0.9*, *r2=0.99*, *L=1.0*, *R1=- 1.5*, *R2=1.5*, *wl=1.064e-06*, *power=False*)

Bases: traits.api.HasTraits

A class to represent a Fabry-Perot cavity.

r1
Input mirror reflectivity (amplitude)

Type float

r2
End mirror reflectivity (amplitude)

Type float

rp1
Input mirror reflectivity (power)

Type float

rp2
End mirror reflectivity (power)

Type float

L
Length

Type float

R1
ROC of the input mirror (positive when concave to incident light, i.e. convex seen from inside the cavity)

Type float

R2
ROC of the end mirror (positive when concave to incident light, i.e. concave seen from inside the cavity)

Type float

wl
Wavelength

Type float

r1

r2

rp1

rp2

tp1

tp2

t1

t2

L

R1

R2

g1

g2

wl

_R1_changed(*self*, *old*, *new*)

_R2_changed(*self*, *old*, *new*)

_L_changed(*self*, *old*, *new*)

_r1_changed(*self*, *old*, *r1*)

_r2_changed(*self*, *old*, *r2*)

finesse(*self*)
Returns the finesse of the cavity.

storageTime(*self*)
Storage time

pole(*self*)
Cavity pole frequency [Hz]

Nbounce(*self*)
Bounce number

powerGain(*self*)
Ratio of the intra-cavity power to the input power.

FSR(*self*)
Returns the free spectral range of the cavity.

modeSpacing(*self*)
Return the transverse mode spacing of the cavity (commonly called gamma). It is a fractional number defined by gamma = (mode spacing frequency)/FSR.

waist(*self*, *size=False*)
Return the q-parameter or the radius of the beam at the cavity waist.

Parameters **size** (*boolean, optional*) – if set to true, the first element of the returned tuple will be the waist size, rather than the q-parameter.

Returns **(q0, d)** – This function returns a tuple with two elements. The first element is the q-parameter of the cavity mode at the cavity waist. If size=True is given, it becomes the waist size (1/e^2 radius). The second element is the distance of the cavity waist from the input mirror.

Return type (complex, float)

spotSize(*self*)
Returns the beam spot sizes on the input and end mirrors as a tuple (w1,w2).

trans(*self*, *f=0*, *d=0*)
Returns the amplitude transmissivity of the cavity. It assumes the cavity was locked to the incident light first. Then computes the amplitude transmissivity for the light with a frequency shift f from the original light with the cavity length changed by d from the initial state.

Parameters

- **f** (*float, optional*) – Frequency shift of the light in Hz. Defaults 0.
- **d** (*float, optional.*) – Cavity length detuning in m. Defaults 0.

Returns The amplitude transmissivity of the cavity (a complex number).

Return type complex

refl(*self*, *f=0*, *d=0*)
Returns the amplitude reflectivity of the cavity. It assumes the cavity was locked to the incident light first. Then computes the amplitude reflectivity for the light with a frequency shift f from the original light with the cavity length changed by d from the initial state.

Parameters

- **f** (*float, optional*) – Frequency shift of the light in Hz. Defaults 0.
- **d** (*float, optional.*) – Cavity length detuning in m. Defaults 0.

Returns The amplitude reflectivity of the cavity (a complex number).

Return type complex

intra(*self*, *f=0*, *d=0*)
Returns the intra cavity field amplitude. It assumes the cavity was locked to the incident light first. Then computes the intra-cavity field amplitude for the light with a frequency shift f from the original light with the cavity length changed by d from the initial state.

Parameters

- **f** (*float, optional*) – Frequency shift of the light in Hz. Defaults 0.
- **d** (*float, optional.*) – Cavity length detuning in m. Defaults 0.

Returns The intra-cavity field amplitude at the input mirror surface (a complex number).

Return type complex

gtrace.optics.cavity.**finesse**(*r1*, *r2*, *power=False*)
Returns the finesse of a cavity

Parameters

- **r1** (*float*) – Reflectivity of the first mirror.
- **r2** (*float*) – Reflectivity of the second mirror.
- **power** (*boolean, optional*) – If True, r1 and r2 are treated as power reflectivities. Otherwise, r1 and r2 are regarded as amplitude reflectivities. Defaults False.

gtrace.optics.consts

Module Contents

Functions

sellmeier(wl, B1, B2, B3, C1, C2, C3)	Calculate index of refraction using Sellmeiers equation
n_fused_silica(wl)	Calculate the index of refraction of fused silica for a given wavelength.
n_sapphire_ordinary(wl)	Calculate the index of refraction of Sapphire ordinary axis for a given wavelength.
n_sapphire_extraordinary(wl)	Calculate the index of refraction of Sapphire extraordinary axis for a given wavelength.

Attributes

n_fused_silica_532nm
n_fused_silica_1064nm
n_fused_silica_1550nm
no_sapphire_532nm
ne_sapphire_532nm
no_sapphire_1064nm
ne_sapphire_1064nm
no_sapphire_1550nm
ne_sapphire_1550nm

gtrace.optics.consts.**sellmeier**(*wl*, *B1*, *B2*, *B3*, *C1*, *C2*, *C3*)
Calculate index of refraction using Sellmeiers equation

n^2 = 1+B1*wl^2/(wl^2 - C1) + B2*wl^2/(wl^2 - C2) + B3*wl^2/(wl^2 - C3)

See below for the coefficients for specific materials. http://www.cvimellesgriot.com/products/Documents/Catalog/Dispersion_Equations.pdf

gtrace.optics.consts.**n_fused_silica**(*wl*)
Calculate the index of refraction of fused silica for a given wavelength.

gtrace.optics.consts.**n_sapphire_ordinary**(*wl*)
Calculate the index of refraction of Sapphire ordinary axis for a given wavelength.

gtrace.optics.consts.**n_sapphire_extraordinary**(*wl*)
Calculate the index of refraction of Sapphire extraordinary axis for a given wavelength.

gtrace.optics.consts.**n_fused_silica_532nm = 1.46071**

gtrace.optics.consts.**n_fused_silica_1064nm = 1.45**

gtrace.optics.consts.**n_fused_silica_1550nm = 1.444**

gtrace.optics.consts.**no_sapphire_532nm = 1.7717**

gtrace.optics.consts.**ne_sapphire_532nm = 1.76355**

gtrace.optics.consts.**no_sapphire_1064nm = 1.75449**

gtrace.optics.consts.**ne_sapphire_1064nm = 1.74663**

gtrace.optics.consts.**no_sapphire_1550nm = 1.74618**

gtrace.optics.consts.**ne_sapphire_1550nm = 1.73838**

gtrace.optics.gaussian

gaussian - Gaussian Optics Module

This module contains several utility functions for gaussian optics.

Module Contents

Functions

modeSpacing(g1, g2)	

q2zr(q)	Convert a q-parameter to Rayleigh range.
q2w(q, wl=1064 * nm)	Convert a q-parameter to the beam size
q2R(q)	Convert a q-parameter to the ROC
Rw2q(ROC=1.0, w=1.0, wl=1.064e-06)	Get the q-parameter from the ROC and w.
InvROCandW2q(invROC=0.0, w=1.0, wl=1.064e-06)	Get the q-parameter from the inverse ROC and w.
zr2w0(zr, wl=1064 * nm)	Convert Rayleigh range to the waist size
w02zr(w0, wl=1064 * nm)	Convert waist size to Rayleigh range
modeMatching(q1, q2x, q2y=False)	Mode matching between two beams with different q-parameters.
modeMatchingElliptic(q1x, q1y, q2x, q2y)	Mode matching between two elliptic beams.
optimalMatching(q1, q2)	Returns a mode (q-parameter) which best matches the given

continues on next page

Table 17 – continued from previous page

qToRadius(q, wl=1.064e-06)	Convert a q-parameter to the beam size
qToROC(q)	Convert a q-parameter to radius of curvature.
ROCandWtoQ(ROC=1.0, w=1.0, wl=1.064e-06)	Convert radius of curvature and beam width to q-parameter
beamClip(a=1.0, w=3.0)	Beam clip
appertureCut(r=1.0, w=3.0)	Apperture cut.

Attributes

pi
sqrt

gtrace.optics.gaussian.**pi**

gtrace.optics.gaussian.**sqrt**

gtrace.optics.gaussian.**modeSpacing**(*g1*, *g2*)

gtrace.optics.gaussian.**q2zr**(*q*)
Convert a q-parameter to Rayleigh range.

Parameters **q** (*complex*) – Beam parameter.

Returns **zr** – Rayleigh range.

Return type float

gtrace.optics.gaussian.**q2w**(*q*, *wl=1064 * nm*)
Convert a q-parameter to the beam size

Parameters

- **q** (*complex*) – Beam parameter.
- **wl** (*float, optional*) – Wavelength. Defaults 1064*nm.

Returns **w** – Beam size.

Return type float

gtrace.optics.gaussian.**q2R**(*q*)
Convert a q-parameter to the ROC

Parameters **q** (*complex*) – Beam parameter.

Returns Radius of curvature.

Return type float

gtrace.optics.gaussian.**Rw2q**(*ROC=1.0*, *w=1.0*, *wl=1.064e-06*)
Get the q-parameter from the ROC and w.

ROC [float, optional] Radius of curvature.

w [float, optional] Beam size. Defaults 1.0.

wl [float, optional] Wavelength. Defaults 1064*nm.

Returns Beam parameter.

Return type complex

gtrace.optics.gaussian.**InvROCandW2q**(*invROC=0.0*, *w=1.0*, *wl=1.064e-06*)
Get the q-parameter from the inverse ROC and w.

Parameters

- **invROC** (*float, optional*) – Inverse of the ROC. Defaults 0.0.
- **w** (*float, optional*) – Beam size. Defaults 1.0.
- **wl** (*float, optional*) – Wavelength. Defaults 1064*nm.

Returns Beam parameter.

Return type complex

gtrace.optics.gaussian.**zr2w0**(*zr*, *wl=1064 * nm*)
Convert Rayleigh range to the waist size

Parameters

- **zr** (*float*) – Rayleigh range.
- **wl** (*float, optional*) – Wavelength. Defaults 1064*nm.

Returns Waist size.

Return type float

gtrace.optics.gaussian.**w02zr**(*w0*, *wl=1064 * nm*)
Convert waist size to Rayleigh range

Parameters

- **w0** (*float*) – Waist size.
- **wl** (*float, optional*) – Wavelength. Defaults 1064*nm.

gtrace.optics.gaussian.**modeMatching**(*q1*, *q2x*, *q2y=False*)
Mode matching between two beams with different q-parameters. The axes of the two beams are assumed to be matched.

Parameters

- **q1** (*complex*) – q-parameter of the first beam. This beam is assumed to be circular.
- **q2x** (*complex*) – q-parameter of the second beam in x-direction. If the second beam is also circular, omit the next argument.
- **q2y** (*complex, optional*) – q-parameter of the second beam in y-direction. Specify this parameter if the second beam is eliptic. Defaults False.

gtrace.optics.gaussian.**modeMatchingElliptic**(*q1x*, *q1y*, *q2x*, *q2y*)
Mode matching between two elliptic beams.

Parameters

- **q1x** (*complex*) – q-parameter of the first beam in x-direction.
- **q1y** (*complex*) – q-parameter of the first beam in y-direction.
- **q2x** (*complex*) – q-parameter of the second beam in x-direction.
- **q2y** (*complex*) – q-parameter of the second beam in y-direction.

Returns

Return type float

gtrace.optics.gaussian.**optimalMatching**(*q1*, *q2*)
Returns a mode (q-parameter) which best matches the given two q-parameters, q1 and q2.

Parameters

- **q1** (*complex*) – q-parameter of the first beam. This beam is assumed to be circular.
- **q2** (*complex*) – q-parameter of the second beam. This beam is assumed to be circular.

Returns

(q, match)

q: The best matching q-parameter

match: Mode matching rate

Return type (complex, match?)

gtrace.optics.gaussian.**qToRadius**(*q*, *wl=1.064e-06*)
Convert a q-parameter to the beam size

Parameters

- **q** (*complex*) – Beam parameter.
- **wl** (*float, optional*) – Wavelength. Defaults 1064e-9.

Returns Radius.

Return type float

gtrace.optics.gaussian.**qToROC**(*q*)
Convert a q-parameter to radius of curvature.

Parameters **q** (*complex*) – Beam parameter.

Returns Radius of curvature.

Return type float

gtrace.optics.gaussian.**ROCandWtoQ**(*ROC=1.0*, *w=1.0*, *wl=1.064e-06*)
Convert radius of curvature and beam width to q-parameter

Parameters

- **ROC** (*float, optional*) – Radius of curvature. Defaults to 1.0.
- **w** (*float, optional*) – Beam width. Defaults to 1.0.
- **wl** (*float, optional*) – Wavelength. Defaults to 1064e-9

Returns q-parameter

Return type complex

gtrace.optics.gaussian.**beamClip**(*a=1.0*, *w=3.0*)
Beam clip

Parameters

- **a** (*float*) –
- **w** (*float*) –

Returns

Return type float

gtrace.optics.gaussian.**appertureCut**(*r=1.0*, *w=3.0*)
Apperture cut.

Parameters

- **r** (*float*) –
- **w** (*float*) –

Returns

Return type float

gtrace.optics.geometric

Module Contents

Functions

deflection_angle(theta, n1, n2, deg=True)	Calculate deflection angle according to Snell's law.
line_plane_intersection(pos, dirVect, plane_center, normalVector, diameter)	Compute the intersection point between a line
line_arc_intersection(pos, dirVect, chord_center, chordNormVect, invROC, diameter, verbose=False)	Compute the intersection point between a line
vector_rotation_2D(vect, angle)	Rotate a 2D vector by an angle.
vector_normalize(vect)	Normalize a vector
normSpheric(normAngle, invROC, dist_from_center)	Returns the local normal angle of a spheric mirror
refl_defl_angle(beamAngle, normAngle, n1, n2, invROC=None)	Returns a tuples of reflection and deflection angles.
cyl_refl_defl_angle(beamAngle, normAngle, n1, n2, invROC=None, curve_direction='h')	Returns a tuples of reflection and deflection angles for incidence of a beam into a cylindrical surface.
vc_deflect(theta, theta1, n1, n2)	Deflection angle helper function for VariCAD.
vc_reflect(theta, theta1)	Convert theta and theta1 to 0-360 format.

Attributes

pi

gtrace.optics.geometric.**pi**

gtrace.optics.geometric.**deflection_angle**(*theta*, *n1*, *n2*, *deg=True*)
Calculate deflection angle according to Snell's law.

Parameters

- **theta** (*float*) – Angle of incidence.
- **n1** (*float*) – Refractive index of the first medium.

- **n2** (*float*) – Refraction index of the second medium.
- **deg** (*boolean, optional*) – True if theta is specified in degrees.

gtrace.optics.geometric.**line_plane_intersection**(*pos*, *dirVect*, *plane_center*, *normalVector*, *diameter*)

Compute the intersection point between a line and a plane

Parameters

- **pos** (*array*) – The position of the end point of the line.
- **dirVert** (*array*) – The directional vector specifying the line.
- **plane_center** (*array*) – The position of the center of the plane.
- **normalVector** (*array*) – The normal vector of the plane.
- **diameter** (*float*) – The diameter of the plane.

Returns The returned value is a dictionary with the following keys: "Intersection Point": numpy array of the coordinates of the intersection point. "isHit": A boolean value of whether the line intersects with the plane or not. "distance": Distance between the origin of the line and the intersection point. "distance from center": Distance between the center of the plane and the intersection point.

Return type dict

gtrace.optics.geometric.**line_arc_intersection**(*pos*, *dirVect*, *chord_center*, *chordNormVect*, *invROC*, *diameter*, *verbose=False*)

Compute the intersection point between a line and an arc.

Parameters

- **pos** (*array*) – Origin of the line.
- **dirVect** (*array*) – Direction of the line.
- **chord_center** (*array*) – The center of the chord made by the arc.
- **chordNormVect** (*array*) – Normal vector of the chord.
- **invROC** (*float*) – Inverse of the ROC of the arc. Positive for concave surface.
- **diameter** (*float*) – Length of the chord.
- **verbose** (*boolean, optional*) – Prints useful information.

Returns The returned value is a dictionary with the following keys: "Intersection Point": numpy array of the coordinates of the intersection point. "isHit": A boolean value of whether the line intersects with the plane or not. "distance": Distance between the origin of the line and the intersection point. "localNormVect": localNormVect, "localNormAngle": localNormAngle.

Return type dict

gtrace.optics.geometric.**vector_rotation_2D**(*vect*, *angle*)

Rotate a 2D vector by an angle.

Parameters

- **vect** (*array*) – A 2D vector.
- **angle** (*float*) – Angle of rotation in radians.

Returns The rotated vector.

Return type array

gtrace.optics.geometric.**vector_normalize**(*vect*)
Normalize a vector

Parameters **vect** (*array*) – The vector to be normalized

Returns The normalized vector.

Return type array

gtrace.optics.geometric.**normSpheric**(*normAngle*, *invROC*, *dist_from_center*)
Returns the local normal angle of a spheric mirror at a distance from the center.

Parameters

- **normAngle** (*float*) – The angle formed by the normal vector of the mirror at the center and the x-axis.
- **invROC** (*float*) – 1/R, where R is the ROC of the mirror.
- **dist_from_center** (*float*) – The distance from the center of the point where the local normal is requested. This is a signed value. For a mirror facing +x (the normal vector points towards positive x direction), this distance is positive for points with positive y coordinate, and negative for points with negative y coordinate.

Returns The local normal angle of a spheric mirror at a distance from the center.

Return type float

gtrace.optics.geometric.**refl_defl_angle**(*beamAngle*, *normAngle*, *n1*, *n2*, *invROC=None*)
Returns a tuples of reflection and deflection angles.

Parameters

- **beamAngle** (*float*) – The angle formed by the propagation direction vector of the incident beam and the x-axis.
- **normAngle** (*float*) – The angle formed by the normal vector of the surface and the x-axis.
- **n1** (*float*) – Index of refraction of the incident side medium.
- **n2** (*float*) – Index of refraction of the transmission side medium.
- **invROC** (*float or None, optional*) – Inverse of the radius of curvature.

Returns

- *6-tuple or 2-tuple*
- *(reflAngle, deflAngle, Mrx, Mry, Mtx, Mty) or (reflAngle, deflAngle)*

gtrace.optics.geometric.**cyl_refl_defl_angle**(*beamAngle*, *normAngle*, *n1*, *n2*, *invROC=None*, *curve_direction='h'*)
Returns a tuples of reflection and deflection angles for incidence of a beam into a cylindrical surface.

Parameters

- **beamAngle** (*float*) – The angle formed by the propagation direction vector of the incident beam and the x-axis.
- **normAngle** (*float*) – The angle formed by the normal vector of the surface and the x-axis.
- **n1** (*float*) – Index of refraction of the incident side medium.
- **n2** (*float*) – Index of refraction of the transmission side medium.
- **invROC** (*float or None, optional*) – Inverse of the radius of curvature.

- **curve_direction** (*str, optional*) – Direction of curvature. Either 'h' or 'v'.

gtrace.optics.geometric.**vc_deflect**(*theta*, *theta1*, *n1*, *n2*)

Deflection angle helper function for VariCAD.

Parameters

- **theta** (*float*) – Angle of the surface measured from right.
- **theta1** (*float*) – Angle of the incident beam measured from right.
- **n1** (*float*) – Index of refraction of the incident side medium.
- **n2** (*float*) – Index of refraction of the transmission side medium.

Returns **phi2** – Angle of the deflected beam measured from right.

Return type float

gtrace.optics.geometric.**vc_reflect**(*theta*, *theta1*)

Convert theta and theta1 to 0-360 format.

Parameters

- **theta** (*float*) – Angle of the surface measured from right.
- **theta1** (*float*) – Angle of the incident beam measured from right.

Returns

Return type float

gtrace.optics.unit

Module Contents

Functions

rad2deg(rad)	
deg2rad(deg)	

Attributes

pi
km
m
cm
mm

continues on next page

Table 22 – continued from previous page

um
nm
kW
W
mW
uW
THz
GHz
MHz
kHz
Hz
mHz
uHz
ppm

gtrace.optics.unit.**pi**

gtrace.optics.unit.**km = 1000.0**

gtrace.optics.unit.**m = 1.0**

gtrace.optics.unit.**cm = 0.01**

gtrace.optics.unit.**mm = 0.001**

gtrace.optics.unit.**um = 1e-06**

gtrace.optics.unit.**nm = 1e-09**

gtrace.optics.unit.**kW = 1000.0**

gtrace.optics.unit.**W = 1.0**

gtrace.optics.unit.**mW = 0.001**

gtrace.optics.unit.**uW = 1e-06**

gtrace.optics.unit.**THz = 1000000000000.0**

gtrace.optics.unit.**GHz = 1000000000.0**

gtrace.optics.unit.**MHz = 1000000.0**

gtrace.optics.unit.**kHz = 1000.0**

gtrace.optics.unit.**Hz = 1.0**

gtrace.optics.unit.**mHz = 0.001**

gtrace.optics.unit.**uHz = 1e-06**

gtrace.optics.unit.**ppm = 1e-06**

gtrace.optics.unit.**rad2deg**(*rad*)

gtrace.optics.unit.**deg2rad**(*deg*)

Package Contents

gtrace.optics.**__author__ = Yoichi Aso**

gtrace.optics.**__copyright__ = Copyright 2011-2021, Yoichi Aso**

gtrace.optics.**__credits__ = ['Yoichi Aso']**

gtrace.optics.**__license__ = BSD**

gtrace.optics.**__version__ = 0.2.1**

gtrace.optics.**__maintainer__ = Yoichi Aso**

gtrace.optics.**__email__ = yoichi.aso@nao.ac.jp**

gtrace.optics.**__status__ = Beta**

5.1.2 Submodules

gtrace.beam

gtrace.beam

A module to define GaussianBeam class.

Module Contents

Classes

GaussianBeam	This is a class to represent a Gaussian beam.

Functions

optFunForStartPointR(phi, Mrot, R, q0, k, sigma, side)	A function to return the distance between the point on
optimStartPointR(theta, R, q0, k, sigma)	Caltulate optimal starting point.
optFunForEndPointR(phi, Mrot, R, q0, k, sigma, side)	A function to return the distance between the point on
optimEndPointR(theta, R, q0, k, sigma)	Caltulate optimal end point.
optFunForFlat(a, Mrot, q0, k, sigma, side)	A function to return the distance between the point on
optimCrossPointFlat(theta, q0, k, sigma)	Caltulate optimal cross point.

Attributes

pi
array
sqrt
__author__
__copyright__
__credits__
__license__
__version__
__maintainer__
__email__
__status__

gtrace.beam.**pi**

gtrace.beam.**array**

gtrace.beam.**sqrt**

gtrace.beam.**__author__ = Yoichi Aso**

gtrace.beam.**__copyright__ = Copyright 2011-2021, Yoichi Aso**

gtrace.beam.**__credits__ = ['Yoichi Aso']**

gtrace.beam.**__license__ = BSD**

gtrace.beam.**__version__ = 0.2.1**

gtrace.beam.**__maintainer__ = Yoichi Aso**

gtrace.beam.**__email__ = yoichi.aso@nao.ac.jp**

gtrace.beam.**__status__ = Beta**

class gtrace.beam.**GaussianBeam**(*q0=1j * 2 * pi / 1064 * nm * 1e-06 / 2*, *q0x=False*, *q0y=False*, *pos=[0.0, 0.0]*, *length=1.0*, *dirAngle=0.0*, *dirVect=None*, *wl=1064 * nm*, *P=1 * W*, *n=1.0*, *name='Beam'*, *layer='main_beam'*)

Bases: traits.api.HasTraits

This is a class to represent a Gaussian beam. A GaussianBeam object has its origin (pos) and a propagation direction (dirVect or dirAngle). A GaussianBeam is characterized by q-parameter(s) at its origin. The beam can be either circular or elliptic. In order to deal with elliptic beams, some parameters are stored in pairs like (q0x, q0y). x and y denote the axes of the cross section of the beam. x-axis is parallel to the paper and the y-axis is perpendicular to the paper.

A beam object can be propagated through a free space or made to interact with an optics.

As a beam propagate through optical system, optical distance and Gouy phase are accumerated.

q
q-parameter of the beam. If the beam is eliptic, q is the q-parameter of the best matching circular mode.

Type complex

qx
q-parameter of the beam in the x-direction.

Type complex

qy
q-parameter of the beam in the y-direction.

Type complex

pos
Position of the beam origin (x, y).

Type array

dirVect
Propagation direction vector.

Type array

dirAngle
Propagation direction angle measured from the positive x-axis.

Type float

length
Length of the beam (used for DXF export)

Type float

layer
Layer name of the beam when exported to a DXF file.

Type str

name
Name of the beam

Type str

wl
Wavelength in vacuum. Not the wavelength in the medium.

Type float

n
Index of refraction of the medium the beam is passing through.

Type float

P
Power.

Type float

wx
Beamwidth in x-direction.

Type float

wy
Beamwidth in y-direction.

Type float

optDist
Accumulated optical distance.

Type float

Gouyx
Accumulated Gouy phase in x-direction.

Type float

Gouyy
Accumulated Gouy phase in y-direction.

Type float

Mx
ABCD matrix in x-direction. This is a 2x2 matrix representing the product of ABCD transformations applied to this beam. It defaults to an identity matrix. Whenever a beam experience an ABCD matrix transformation, such as propagation in the space or reflection by a curved mirror, the applied ABCD matrix is multiplied to this matrix, so that we can keep track of what kind of transformations were made during beam propagation.

Type array

My
ABCD matrix in y-direction. The meaning is the same as Mx.

Type array

departSurfAngle
The angle formed by x-axis and the normal vector of the surface from which the beam is departing. Default is None. Used by the drawing routine.

Type None

departSurfInvROC
Inverse of the ROC of the surface from which the beam is departing. The ROC is positive for a concave surface seen from the beam side. Default is None. Used by the drawing routine.

Type None

incSurfAngle
The angle formed by the x-arm and the normal vector of the surface to which the beam is incident. Default is None. Used by the drawing routine.

Type None

incSurfInvROC
Inverse of the ROC of the surface to which the beam is incident. The ROC is positive for a concave surface seen from the beam side. Default is None. Used by the drawing routine.

Type None

stray_order
An integer indicating if this beam is a stray light or not. The default value is 0. Every time a beam is reflected by an AR surface or transmits an HR surface, this couter is increased by 1.

Type int

name

wl

P

q

qx

qy

qrx

qry

Gouyx

Gouyy

wx

wy

n

pos

length

layer

dirVect

dirAngle

optDist

Mx

My

copy(*self*)
Make a deep copy.

propagate(*self*, *d*)
Propagate the beam by a distance d from the current position. self.n is used as the index of refraction. During this process, the optical distance traveled is added to self.optDist. self.Goux and self.Gouyy are also updated to record the Gouy phase change.

Parameters **d** (*float*) – Distance.

ABCDTrans(*self*, *ABCDx*, *ABCDy=None*)
Apply ABCD transformation to the beam.

Parameters

- **ABCDx** (*array*) – ABCD matrix for x-direction.
- **ABCDy** (*array or None, optional.*) – ABCD matrix for y-direction. Defaults None. If None, set to ABCDx.

rotate(*self*, *angle*, *center=False*)
Rotate the beam around 'center'. If center is not given, the beam is rotated around self.pos.

Parameters

- **angle** (*float*) – Rotation angle in radians.

- **center** (*array or boolean.*) – Center of rotation. Should be an array of shape(2,). Defaults False.

translate (*self*, *trVect*)
Translate the beam by the direction and the distance specified by a vector.

Parameters **trVect** (*array*) – A vector to specify the translation direction and distance. Should be an array of shape(2,)

flip (*self*, *flipDirVect=True*)
Change the propagation direction of the beam by 180 degrees. This is equivalent to the reflection of the beam by a spherical mirror with the same ROC as the beam.

If optional argument flipDirVect is set to False, the propagation direction of the beam is not changed.

Parameters **flipDirVect** (*boolean, optional*) – Flip propagation direction. Defaults True.

width (*self*, *dist*)
Returns the beam width at a distance dist from the origin of the beam. The width is the radius where the light power becomes 1/e^2.

Parameters **dist** (*float*) – Distance.

Returns The width of the beam in x and y direction.

Return type (float, float)

R (*self*, *dist=0.0*)
Returns the beam ROC at a distance dist from the origin of the beam.

Parameters **dist** (*float, optional*) – Distance.

Returns Beam ROC.

Return type (float, float)

waist (*self*)
Return the tuples of waist size and distance

Returns {"Waist Size": (float, float), "Waist Position": (float, float)}

Return type dict

draw (*self*, *cv*, *sigma=3.0*, *mode='x'*, *drawWidth=True*, *fontSize=False*, *drawPower=False*, *drawROC=False*, *drawGouy=False*, *drawOptDist=False*, *drawName=False*, *debug=False*)
Draw the beam into a DXF object.

Parameters

- **cv** (gtrace.draw.draw.Canvas) – gtrace canvas.
- **sigma** (*float, optional*) – The width of the beam drawn is sigma * (1/e^2 radius of the beam). The default is sigma = 3. sigma = 2.7 gives 1ppm diffraction loss. Defaults 3.
- **mode** (*str, optional*) – 'avg', 'x', or 'y'. A beam can have different widths for x- and y- directions. If 'avg' is specified, the average of them are drawn. 'x' and 'y' specifies to show the width of the respective directions. Defaults 'x'.
- **fontSize** (*float, optional*) – Size of the font used to show supplemental informations. Defaults False.
- **drawWidth** (*boolean, optional*) – Whether to draw width or not. Defaults True.

- **drawPower** (*boolean, optional*) – Whether to show the beam power. Defaults False.
- **drawROC** (*boolean, optional*) – Whether to show the ROC or not. Defaults False.
- **drawGouy** (*boolean, optional*) – Whether to show the Gouy phase or not. Defaults False.
- **drawOptDist** (*boolean, optional*) – Whether to show the accumulated optical distance or not. Defaults False.
- **drawName** (*boolean, optional*) – Whether draw the name of the beam or not. Defaults False.
- **debug** (*boolean, optional*) – Debug.

drawWidth(*self*, *cv*, *sigma*, *mode*)

Draw width on canvas.

Parameters

- **cv** (gtrace.draw.draw.Canvas) – The canvas.
- **sigma** (*float*) – The width of the beam drawn is sigma * (1/e^2 radius of the beam). The default is sigma = 3. sigma = 2.7 gives 1ppm diffraction loss.
- **mode** (*str*) – 'avg', 'x', or 'y'. A beam can have different widths for x- and y- directions. If 'avg' is specified, the average of them are drawn. 'x' and 'y' specifies to show the width of the respective directions.

_dirAngle_changed(*self*, *old*, *new*)

_dirVect_changed(*self*, *old*, *new*)

_qx_changed(*self*, *old*, *new*)

_qy_changed(*self*, *old*, *new*)

_qrx_changed(*self*, *old*, *new*)

_qry_changed(*self*, *old*, *new*)

_n_changed(*self*, *old*, *new*)

gtrace.beam.**optFunForStartPointR**(*phi*, *Mrot*, *R*, *q0*, *k*, *sigma*, *side*)

A function to return the distance between the point on the spherical surface at an angle phi and the beam width at the same z.

Parameters

- **phi** (*float*) – phi
- **Mrot** (*array*) – Rotational transformation.
- **R** (*float*) – R
- **q0** (*complex*) – Beam parameter.
- **k** (*float*) – k
- **sigma** (*float*) – Beam width
- **side** – side

Returns Distance between the point on the spherical surface at an angle phi and the beam width at the same z.

Return type float

gtrace.beam.**optimStartPointR**(*theta*, *R*, *q0*, *k*, *sigma*)
Caltulate optimal starting point.

theta [float] theta

R [float] R

q0 [complex] Beam parameter.

k [float] k

sigma [float] Beam width.

Returns Optimal starting point.

Return type (float, float)

gtrace.beam.**optFunForEndPointR**(*phi*, *Mrot*, *R*, *q0*, *k*, *sigma*, *side*)
A function to return the distance between the point on the spherical surface at an angle phi and the beam width at the same z.

Parameters

- **phi** (*float*) – phi
- **Mrot** (*array*) – Rotational transformation.
- **R** (*float*) – R
- **q0** (*complex*) – Beam parameter.
- **k** (*float*) – k
- **sigma** (*float*) – Beam width
- **side** – side

Returns Distance between the point on the spherical surface at an angle phi and the beam width at the same z.

Return type float

gtrace.beam.**optimEndPointR**(*theta*, *R*, *q0*, *k*, *sigma*)
Caltulate optimal end point.

theta [float] theta

R [float] R

q0 [complex] Beam parameter.

k [float] k

sigma [float] Beam width.

Returns Optimal end point.

Return type (float, float)

gtrace.beam.**optFunForFlat**(*a*, *Mrot*, *q0*, *k*, *sigma*, *side*)
A function to return the distance between the point on the spherical surface (flat?) at an angle phi and the beam width at the same z.

Parameters

- **a** (*float*) – a

- **Mrot** (*array*) – Rotational transformation.
- **q0** (*complex*) – Beam parameter.
- **k** (*float*) – k
- **sigma** (*float*) – Beam width
- **side** – side

Returns Distance between the point on the spherical surface at an angle phi and the beam width at the same z.

Return type float

gtrace.beam.**optimCrossPointFlat**(*theta*, *q0*, *k*, *sigma*)

Caltulate optimal cross point.

theta [float] theta

R [float] R

q0 [complex] Beam parameter.

k [float] k

sigma [float] Beam width.

Returns Optimal end point.

Return type (float, float)

gtrace.nonsequential

gtrace.nonsequential

A module to perform non-sequential trace of a beam in an optical system.

Module Contents

Functions

non_seq_trace(optList, src_beam, order=10, power_threshold=0.1, open_beam_length=1.0)	Perform non-sequential trace of the source beam, src_beam,

Attributes

__author__
__copyright__
__credits__
__license__

continues on next page

Table 27 – continued from previous page

__version__
__maintainer__
__email__
__status__

gtrace.nonsequential.**__author__ = Yoichi Aso**

gtrace.nonsequential.**__copyright__ = Copyright 2011-2021, Yoichi Aso**

gtrace.nonsequential.**__credits__ = ['Yoichi Aso']**

gtrace.nonsequential.**__license__ = BSD**

gtrace.nonsequential.**__version__ = 0.2.1**

gtrace.nonsequential.**__maintainer__ = Yoichi Aso**

gtrace.nonsequential.**__email__ = yoichi.aso@nao.ac.jp**

gtrace.nonsequential.**__status__ = Beta**

gtrace.nonsequential.**non_seq_trace**(*optList*, *src_beam*, *order=10*, *power_threshold=0.1*, *open_beam_length=1.0*)

Perform non-sequential trace of the source beam, src_beam, through the optical system represented by a collection of optics, optList.

Parameters

- **optList** (*list of gtrace.optcomp.Optics*) – List of optical components.
- **src_beam** (gtrace.beam.GaussianBeam) – The source beam object.
- **order** (*int, optional*) – An integer to specify how many times the internal reflections are computed. Defaults to 10.
- **power_threshold** (*float, optional*) – The power threshold for internal reflection calculation. If the power of an auxiliary beam falls below this threshold, further propagation of this beam will not be performed. Defaults to 0.1.
- **open_beam_length** (*float, optional*) – The default length for beams that are not hitting anything. Defaults to 1.0.

Returns **terminated_beam_list** – A list of beams.

Return type list of gtrace.beam.GaussianBeam

gtrace.optcomp

Define optical components for gtrace.

Module Contents

Classes

Optics	A general optics class from which other specific
Mirror	Representing a partial reflective mirror.
CyMirror	Representing a partial reflective cylindrical mirror. Note that both HR and AR surfaces are treated as cylindrical if you specify non-zero ROC for them. The curve directions of the two surfaces must be the same.

Attributes

pi
array
sqrt
__author__
__copyright__
__credits__
__license__
__version__
__maintainer__
__email__
__status__

gtrace.optcomp.**pi**

gtrace.optcomp.**array**

gtrace.optcomp.**sqrt**

gtrace.optcomp.**__author__ = Yoichi Aso**

gtrace.optcomp.**__copyright__ = Copyright 2011-2021, Yoichi Aso**

gtrace.optcomp.**__credits__ = ['Yoichi Aso']**

gtrace.optcomp.**__license__ = BSD**

gtrace.optcomp.**__version__ = 0.2.1**

gtrace.optcomp.**__maintainer__ = Yoichi Aso**

gtrace.optcomp.**__email__ = yoichi.aso@nao.ac.jp**

gtrace.optcomp.**__status__ = Beta**

class gtrace.optcomp.**Optics**
Bases: traits.api.HasTraits

A general optics class from which other specific optics classes are derived.

name
Name of the optics.

Type str

center
Center position of the optics. array of shape(2,).

Type array

rotationAngle
This angle defines the orientation of the optics.

Type float

name

center

rotationAngle

isHit(*beam*)
A function to see if a beam hits this optics or not.

Parameters **beam** (gtrace.beam.GaussianBeam) – A GaussianBeam object to be interacted by the optics.

Returns

The return value is a dictionary with the following keys: `isHit, position, distance, face`

`isHit`: This is a boolean to answer whether the beam hit the optics or not.

`position`: A numpy array containing the coordinate values of the intersection point between the beam and the optics. If isHit is False, this parameter does not mean anything.

`distance` The distance between the beam origin and the intersection point.

`face`: An optional string identifying which face of the optics was hit. For example, `face` can be either "HR" or "AR" for a mirror. `face` can also be "side", meaning that the beam hits a side of the optics, which is not meant to be used, e.g. the side of a mirror. In this case, the beam have reached a dead end.

Return type Dict

hit(*beam*, *order=0*, *threshold=0.0*)
A function to hit the optics with a beam.

This function attempts to hit the optics with the source beam, `beam`.

Parameters

- **beam** (gtrace.beam.GaussianBeam) – A GaussianBeam object to be interacted by the optics.
- **order** (*int, optional*) – An integer to specify how many times the internal reflections are computed. Defaults 0.

- **threshold** (*float, optional*) – The power threshold for internal reflection calculation. If the power of an auxiliary beam falls below this threshold, further propagation of this beam will not be performed. Defaults 0.0.

Returns

(isHit, beamDict, face)

isHit This is a boolean to answer whether the beam hit the optics or not.

beamDict A dictionary containing resultant beams.

face: An optional string identifying which face of the optics was hit. For a mirror, face is any of "HR", "AR" or "side".

Return type {boolean, dict, str}

isHitSurface(*self*, *beam*, *surface_center*, *normal_vector*, *surface_size=1.0*, *inv_ROC=0.0*)

Determine if a beam hit a surface

Parameters **beam** (gtrace.beam.GaussianBeam) – A GaussianBeam object to be interacted by the optics.

Returns

ans – A dictionary with the following keys: "isHit": A boolean value whether the beam hit the surface or not. "Intersection Point": numpy array of the coordinates of the intersection point. "distance": Distance between the origin of the beam and the intersection point. "localNormVect": A numpy array representing the normal vector of the surface

at the intersection point.

"localNormAngle": The angle of the localNormVect.

Return type dict

class gtrace.optcomp.**Mirror**(*HRcenter=[0.0, 0.0]*, *normAngleHR=0.0*, *normVectHR=None*, *diameter=25.0 * cm*, *thickness=15.0 * cm*, *wedgeAngle=0.25 * pi / 180.0*, *inv_ROC_HR=1.0 / 7000.0*, *inv_ROC_AR=0.0*, *Refl_HR=0.99*, *Trans_HR=0.01*, *Refl_AR=0.01*, *Trans_AR=0.99*, *n=1.45*, *name='Mirror'*, *HRtransmissive=False*, *term_on_HR=False*)

Bases: *Optics*

Representing a partial reflective mirror.

curve_direction

Either 'h' or 'v'. If it is 'h' the mirror is curved in horizontal plane. If 'v', it is vertical.

Type str

HRcenter

The position of the center of the arc of the HR surface. shape(2,).

Type array

HRcenterC

The position of the center of the chord of the HR surface. shape(2,).

Type array

normVectHR

Normal vector of the HR surface. shape(2,)

Type array

normAngleHR
Angle of the HR normal vector. In radians.

Type float

ARcenter
The position of the center of the AR surface. shape(2,)

Type array

normVectAR
Normal vector of the HR surface. shape(2,)

Type array

normAngleAR
Angle of the HR normal vector. In radians.

Type float

HRtransmissive
A boolean value defaults to False. If True, this mirror is supposed to transmit beams on the HR surface. Therefore, for the first encounter of a beam on the HR surface of this mirror will not increase the stray_order. This flag should be set to True for beam splitters and input test masses.

Type boolean

term_on_HR
If this is True, a beam with stray_order <= self.term_on_HR_order will be terminated when it hits on HR. This is to avoid the inifinite loop of non-sequencial trace by forming a cavity.

Type boolean

term_on_HR_order
Integer to specify the upper limit of the stray order used to judge whether to terminate the non sequential trace or not on HR reflection.

Type int

HRcenter

HRcenterC

sagHR

normVectHR

normAngleHR

ARcenter

ARcenterC

sagAR

normVectAR

normAngleAR

diameter

ARdiameter

thickness

wedgeAngle

n

inv_ROC_HR

inv_ROC_AR

Refl_HR

Trans_HR

Refl_AR

Trans_AR

copy(*self*)

get_side_info(*self*)
Return information on the sides of the mirror. Returned value is a list of two tuples like [(center1, normVect1, length1), (center2, normVect2, length2)] Each tuple corresponds to a side. center1 is the coordinates of the center of the side line. normVect1 is the normal vector of the side line. length1 is the length of the side line.

Returns

Return type [(float, float, float), (float, float, float)]

rotate(*self*, *angle*, *center=False*)
Rotate the mirror. If center is not specified, the center of rotation is HRcenter. If center is given (as a vector), the center of rotation is center. center is a position vector in the global coordinates.

Parameters

- **angle** (*float*) – Angle of rotation.
- **center** (*array or boolean, optional*) – Center of rotation, or False.

translate(*self*, *trVect*)

draw(*self*, *cv*, *drawName=False*)
Draw itself

isHit(*self*, *beam*)
A function to see if a beam hits this optics or not.

Parameters **beam** (gtrace.beam.GaussianBeam) – A GaussianBeam object to be interacted by the optics.

Returns

The return value is a dictionary with the following keys: `isHit`, `position`, `distance`, `face`

`isHit`: This is a boolean to answer whether the beam hit the optics or not.

`position`: A numpy array containing the coordinate values of the intersection point between the beam and the optics. If isHit is False, this parameter does not mean anything.

`distance` The distance between the beam origin and the intersection point.

`face`: An optional string identifying which face of the optics was hit. For example, `face` can be either "HR" or "AR" for a mirror. `face` can also be "side", meaning that the beam hits a side of the optics, which is not meant to be used, e.g. the side of a mirror. In this case, the beam have reached a dead end.

Return type Dict

hit (*self*, *beam*, *order=0*, *threshold=0.0*, *face=False*)
A function to hit the optics with a beam.

This function attempts to hit the optics with the source beam, `beam`.

Parameters

- **beam** (`gtrace.beam.GaussianBeam`) – A GaussianBeam object to be interacted by the optics.
- **order** (*int, optional*) – An integer to specify how many times the internal reflections are computed. Defaults 0.
- **threshold** (*float, optional*) – The power threshold for internal reflection calculation. If the power of an auxiliary beam falls below this threshold, further propagation of this beam will not be performed. Defaults 0.0.

Returns

`(isHit, beamDict, face)`

`isHit` This is a boolean to answer whether the beam hit the optics or not.

`beamDict` A dictionary containing resultant beams.

face: An optional string identifying which face of the optics was hit. For a mirror, `face` is any of "HR", "AR" or "side".

Return type {boolean, dict, str}

hitFromHR (*self*, *beam*, *order=0*, *threshold=0.0*, *verbose=False*)
Compute the reflected and deflected beams when an input beam hit the HR surface.

The internal reflections are computed as long as the number of internal reflections are below the `order` and the power of the reflected beams is over the threshold.

Parameters

- **beam** (`gtrace.beam.GaussianBeam`) – A GaussianBeam object to be interacted by the optics.
- **order** (*int, optional*) – An integer to specify how many times the internal reflections are computed. Defaults 0.
- **threshold** (*float, optional*) – The power threshold for internal reflection calculation. If the power of an auxiliary beam falls below this threshold, further propagation of this beam will not be performed. Defaults 0.0.
- **verbose** (*boolean, optional*) – Print useful information.

Returns **beams** – Dictionary of reflected and deflected beams.

Return type dict

hitFromAR (*self*, *beam*, *order=0*, *threshold=0.0*, *verbose=False*)
Compute the reflected and deflected beams when an input beam hit the AR surface.

The internal reflections are computed as long as the number of internal reflections are below the `order` and the power of the reflected beams is over the threshold.

Parameters

- **beam** (`gtrace.beam.GaussianBeam`) – A GaussianBeam object to be interacted by the optics.

- **order** (*int, optional*) – An integer to specify how many times the internal reflections are computed. Defaults 0.
- **threshold** (*float, optional*) – The power threshold for internal reflection calculation. If the power of an auxiliary beam falls below this threshold, further propagation of this beam will not be performed. Defaults 0.0.
- **verbose** (*boolean, optional*) – Print useful information.

Returns **beams** – Dictionary of reflected and deflected beams.

Return type dict

_normAngleHR_changed(*self*, *old*, *new*)

_normVectHR_changed(*self*, *old*, *new*)

_HRcenterC_changed(*self*, *old*, *new*)

_HRcenter_changed(*self*, *old*, *new*)

_center_changed(*self*, *old*, *new*)

_wedgeAngle_changed(*self*, *old*, *new*)

_inv_ROC_HR_changed(*self*, *old*, *new*)

_inv_ROC_AR_changed(*self*, *old*, *new*)

class gtrace.optcomp.**CyMirror**(*HRcenter=[0.0, 0.0]*, *normAngleHR=0.0*, *normVectHR=None*, *diameter=25.0 * cm*, *thickness=15.0 * cm*, *wedgeAngle=0.25 * pi / 180.0*, *inv_ROC_HR=1.0 / 7000.0*, *inv_ROC_AR=0.0*, *Refl_HR=0.99*, *Trans_HR=0.01*, *Refl_AR=0.01*, *Trans_AR=0.99*, *n=1.45*, *name='Mirror'*, *HRtransmissive=False*, *term_on_HR=False*, *curve_direction='h'*)

Bases: *Mirror*

Representing a partial reflective cylindrical mirror. Note that both HR and AR surfaces are treated as cylindrical if you specify non-zero ROC for them. The curve directions of the two surfaces must be the same.

curve_direction
Either 'h' or 'v'. If it is 'h' the mirror is curved in horizontal plane. If 'v', it is vertical.

Type str

HRcenter
The position of the center of the arc of the HR surface. shape(2,).

Type array

HRcenterC
The position of the center of the chord of the HR surface. shape(2,).

Type array

normVectHR
Normal vector of the HR surface. shape(2,)

Type array

normAngleHR
Angle of the HR normal vector. In radians.

Type float

ARcenter
The position of the center of the AR surface. shape(2,)

Type array

normVectAR
Normal vector of the HR surface. shape(2,)

Type array

normAngleAR
Angle of the HR normal vector. In radians.

Type float

HRtransmissive
A boolean value defaults to False. If True, this mirror is supposed to transmit beams on the HR surface. Therefore, for the first encounter of a beam on the HR surface of this mirror will not increase the stray_order. This flag should be set to True for beam splitters and input test masses.

Type boolean

term_on_HR
If this is True, a beam with stray_order <= self.term_on_HR_order will be terminated when it hits on HR. This is to avoid the inifinite loop of non-sequencial trace by forming a cavity.

Type boolean

term_on_HR_order
Integer to specify the upper limit of the stray order used to judge whether to terminate the non sequential trace or not on HR reflection.

Type int

copy(*self*)

get_side_info(*self*)
Return information on the sides of the mirror. Returned value is a list of two tuples like [(center1, normVect1, length1), (center2, normVect2, length2)] Each tuple corresponds to a side. center1 is the coordinates of the center of the side line. normVect1 is the normal vector of the side line. length1 is the length of the side line.

Returns

Return type [(float, float, float), (float, float, float)]

isHit(*self*, *beam*)
A function to see if a beam hits this optics or not.

Parameters **beam** (gtrace.beam.GaussianBeam) – A GaussianBeam object to be interacted by the optics.

Returns

The return value is a dictionary with the following keys: `isHit, position, distance, face`

`isHit`: This is a boolean to answer whether the beam hit the optics or not.

`position`: A numpy array containing the coordinate values of the intersection point between the beam and the optics. If isHit is False, this parameter does not mean anything.

`distance` The distance between the beam origin and the intersection point.

`face`: An optional string identifying which face of the optics was hit. For example, `face` can be either "HR" or "AR" for a mirror. `face` can also be "side", meaning that the beam hits a side of the optics, which is not meant to be used, e.g. the side of a mirror. In this case, the beam have reached a dead end.

Return type Dict

draw (*self*, *cv*, *drawName=False*)
Draw itself

hitFromHR (*self*, *beam*, *order=0*, *threshold=0.0*, *verbose=False*)
Compute the reflected and deflected beams when an input beam hit the HR surface.

The internal reflections are computed as long as the number of internal reflections are below the `order` and the power of the reflected beams is over the threshold.

Parameters

- **beam** (`gtrace.beam.GaussianBeam`) – A GaussianBeam object to be interacted by the optics.
- **order** (*int, optional*) – An integer to specify how many times the internal reflections are computed. Defaults 0.
- **threshold** (*float, optional*) – The power threshold for internal reflection calculation. If the power of an auxiliary beam falls below this threshold, further propagation of this beam will not be performed. Defaults 0.0.
- **verbose** (*boolean, optional*) – Print useful information.

Returns **beams** – Dictionary of reflected and deflected beams.

Return type dict

hitFromAR (*self*, *beam*, *order=0*, *threshold=0.0*, *verbose=False*)
Compute the reflected and deflected beams when an input beam hit the AR surface.

The internal reflections are computed as long as the number of internal reflections are below the `order` and the power of the reflected beams is over the threshold.

Parameters

- **beam** (`gtrace.beam.GaussianBeam`) – A GaussianBeam object to be interacted by the optics.
- **order** (*int, optional*) – An integer to specify how many times the internal reflections are computed. Defaults 0.
- **threshold** (*float, optional*) – The power threshold for internal reflection calculation. If the power of an auxiliary beam falls below this threshold, further propagation of this beam will not be performed. Defaults 0.0.
- **verbose** (*boolean, optional*) – Print useful information.

Returns **beams** – Dictionary of reflected and deflected beams.

Return type dict

gtrace.unit

Module Contents

Functions

rad2deg(rad)	
deg2rad(deg)	

Attributes

pi	
__author__	
__copyright__	
__credits__	
__license__	
__version__	
__maintainer__	
__email__	
__status__	
km	
m	
cm	
mm	
um	
nm	
kW	
W	
mW	

continues on next page

Table 31 – continued from previous page

uW
THz
GHz
MHz
kHz
Hz
mHz
uHz
ppm
percent

gtrace.unit.**pi**

gtrace.unit.**__author__ = Yoichi Aso**

gtrace.unit.**__copyright__ = Copyright 2011, Yoichi Aso**

gtrace.unit.**__credits__ = ['Yoichi Aso']**

gtrace.unit.**__license__ = BSD**

gtrace.unit.**__version__ = 0.2.1**

gtrace.unit.**__maintainer__ = Yoichi Aso**

gtrace.unit.**__email__ = yoichi.aso@nao.ac.jp**

gtrace.unit.**__status__ = Beta**

gtrace.unit.**km = 1000.0**

gtrace.unit.**m = 1.0**

gtrace.unit.**cm = 0.01**

gtrace.unit.**mm = 0.001**

gtrace.unit.**um = 1e-06**

gtrace.unit.**nm = 1e-09**

gtrace.unit.**kW = 1000.0**

gtrace.unit.**W = 1.0**

gtrace.unit.**mW = 0.001**

gtrace.unit.**uW = 1e-06**

gtrace.unit.**THz = 1000000000000.0**

gtrace.unit.**GHz = 1000000000.0**

gtrace.unit.**MHz = 1000000.0**

gtrace.unit.**kHz = 1000.0**

gtrace.unit.**Hz = 1.0**

gtrace.unit.**mHz = 0.001**

gtrace.unit.**uHz = 1e-06**

gtrace.unit.**ppm = 1e-06**

gtrace.unit.**percent = 0.01**

gtrace.unit.**rad2deg**(*rad*)

gtrace.unit.**deg2rad**(*deg*)

5.1.3 Package Contents

gtrace.**__author__ = Yoichi Aso**

gtrace.**__copyright__ = Copyright 2011-2021, Yoichi Aso**

gtrace.**__credits__ = ['Yoichi Aso']**

gtrace.**__license__ = BSD**

gtrace.**__version__ = 0.2.1**

gtrace.**__maintainer__ = Yoichi Aso**

gtrace.**__email__ = yoichi.aso@nao.ac.jp**

gtrace.**__status__ = Beta**

CHAPTER

SIX

INDICES AND TABLES

- genindex
- modindex
- search

PYTHON MODULE INDEX

g

INDEX

Symbols

A

B

C

D

E

F

G

H

I

K

L

M

N

O

P

Q

R

S

T

U

V

W

Z