Paraxial Ray Tracing in Python

David Schmidt

February 25, 2018

Outline

Introduction

Input Data for Lens System

Ray Start Data

- Marginal and Chief Ray
- N Rays

Paraxial Ray Tracing Equations

Figure Generation

Plot Lenses

Animation of Rays

- Marginal and Chief Ray
- N Rays

Examples

Housekeeping

```
11  import numpy as np
12  import matplotlib.pyplot as plt
13  import matplotlib as mpl
14  import matplotlib.animation as animation
15
16  #Housekeeping to set plot parameters and close any open figure
17  plt.close()
18  font = {'weight' : 'bold',
19          'size'   : 22}
20
21  mpl.rc('font', **font)
22
```

Input Data for Lens System

```
23  #%% Lens Data (input as surfaces)
24  """
25  Area to input lens system paramters. Two examples are given to show how to make the system.
26  Assumes even number of surfaces.
27  """
28  surf_pow=[0.015071,0,-0.011662,-0.016327,0.005736,0.01424,0.02,0.01]
29  surf_dis=[40,8.74,11.05,2.78,7.63,9.54,20,4,100]
30  surf_ind=[1,1.617,1,1.649,1,1.617,1,3,1]
31  surf_tot=8
32
33  lenstot=int(surf_tot/2)
34  lens_pos=[]
35  for ii in range(lenstot):
36      lens_pos.append([sum(surf_dis[0:(ii*2+1)]),30,surf_dis[ii*2+1],surf_ind[ii*2+1]])
37
38  """
39  surf_pow=[0.00125,0.00125]
40  surf_dis=[10,4,450]
41  surf_ind=[1,1.516824,1]
42  surf_tot=2
43
44  lenstot=1
45  lens_pos=[[surf_dis[0],40,surf_dis[1]]]
46  """
```

Marginal and Chief Ray

```
47  #%% Ray data [marginal ray, chief ray]
48  h=[14.93368,-14*np.pi/180*surf_dis[0]-6.9574]
49  ang=[0,14*np.pi/180]
50
51  #h=[20,-5*np.pi/180*surf_dis[0]]
52  #ang=[0,5*np.pi/180]
53
54
```

N Rays

```
47  #%% Ray data
48
49  #Ray data is populated below by giving amount of rays and an angle
50
51  nray_m=80
52  nray_a=80
53  angle=5
54  h_m=np.linspace(-20,20,nray_m)
55  h_a=np.linspace(-angle*np.pi/180*surf_dis[0]-20,-angle*np.pi/180*surf_dis[0]+20,nray_a)
56  ang_m=np.zeros(nray_m)
57  ang_a=np.ones(nray_a)*angle*np.pi/180
58
59  tot=len(h_m)+len(h_a)
60  h=np.reshape([h_m,h_a],(1,tot))[0]
61  ang=np.reshape([ang_m,ang_a],(1,tot))[0]
62
63
```

Paraxial Ray Tracing Equations

```
55  #%% paraxial equations
56
57  def refract(nin,nout,ang,y,lpow):
58      return (nin*ang-y*lpow)/nout
59
60  def prop(y,ang,dis):
61      return (ang*dis+y)
62
```

Figure Generation

```
93    xlimlow=0
94    xlimhigh=sum(surf_dis)+5
95    ylimlow=-(max(h)+25)
96    ylimhigh=-ylimlow
97
98    fig = plt.figure(figsize=(20, 10), dpi=80, facecolor='w', edgecolor='k')
99    ax = fig.add_subplot(111, aspect='equal', xlim=(xlimlow, xlimhigh), ylim=(ylimlow,
       ↪    ylimhigh),ylabel="Ray height (mm)",xlabel="Distance (mm)",title="Ray Tracing System")
100   ax.grid()
101
```

Plot Lenses

```
102     #%% Place paraxial lenses
103     for ii in range(0,lenstot):
104         lens_use=lens_pos[ii]
105         xl=lens_use[0]
106         yl=lens_use[1]
107         wl=lens_use[2]
108         ax.add_patch(mpl.patches.Rectangle((xl,-yl), width=wl, height=2*yl, angle=0.0,
              ↪    color='b',alpha=lens_use[3]/max(surf_ind)))
109
110
```

Marginal and Chief Ray

```
111  #%% generate lines for animation from ray plot
112  plotlays, plotcols = [2], ["black","red"]
113  lines = []
114  for index in range(2):
115      lobj = ax.plot([],[],lw=2,color=plotcols[index])[0]
116      lines.append(lobj)
117
118  def init():
119      for line in lines:
120          line.set_data([],[])
121      return lines
122
123  def animate(i):
124      xu = [x[0:i],x[0:i]]
125      yu = [y_marg[0:i],y_chief[0:i]]
126      for lnum,line in enumerate(lines):
127          line.set_data(xu[lnum], yu[lnum]) # set data for each line separately.
128      return lines
129
130  ani = animation.FuncAnimation(fig, animate, interval=1, blit=True, init_func=init)
131
132  plt.show()
```

N Rays

```
114  #%% generate lines for animation from ray plot
115  plotcols=["blue"]*nray_m+["red"]*nray_a
116
117  lines = []
118  for index in range(rays):
119      lobj = ax.plot([],[],lw=2,color=plotcols[index])[0]
120      lines.append(lobj)
121
122  def init():
123      for line in lines:
124          line.set_data([],[])
125      return lines
126
127  def animate(i):
128      xu=[]
129      yu=[]
130      for ll in range(rays):
131          xu.append(x[0:i])
132          yu.append(y_set[0:i,ll])
133      for lnum,line in enumerate(lines):
134          line.set_data(xu[lnum], yu[lnum]) # set data for each line separately.
135      return lines
136
137  ani = animation.FuncAnimation(fig, animate, interval=1, blit=True, init_func=init)
138
139  plt.show()
```

Examples

Lets go see the code in action