LightPipes: software for education in coherent optics.

Gleb Vdovin,
Electronic Instrumentation, TU Delft
P.O. Box 5031, 2600 GA Delft The Netherlands,
Phone 31-15-2786165, Fax 31-15-2785755, email gleb@ei.et.tudelft.nl
Hedser van Brug
Optics Research Group, Applied Physics,
Delft University of Technology Lorentzweg 1
2628 CJ Delft The Netherlands Phone : +31 15 2784288 Fax : +31 15 2788105
Fred van Goor
Quantum Electronics Group, Applied Physics University of Twente
P.O. Box 217, 7500 AE Enschede, The Netherlands.
Phone +31 53 4893963, Fax +31 53 4891102, e-mail F.A.vanGoor@tn.utwente.nl

Keywords: optical software, wave optics, diffraction, interference

ABSTRACT

LightPipes is a portable set of software tools, designed to model the propagation of coherent light in optical systems using a numerical approximations of the scalar theory of diffraction. Models of interferometers, laser resonators, waveguides, holographic setups and many more can be easily built using the components of the package. The command-line shell-based version of the toolbox written originally in C works under Unix, MSDOS and MAC OS. The toolbox also exists as a stand-alone program written in C++, which can be compiled virtually under any existing operational system. A MathCad port of LightPipes provides an extended user interface and full compatibility with the MathCad environment, the Java port allows for running the package over the net using the Netscape web browser.

1. GENERAL CONCEPTS

The LightPipes toolbox is based on the concept of sharing a huge data structure, describing the distribution of the scalar intensity and phase in the coherent light beam by a set of processing software tools. Each tool has a standard interface to read and to write the structured data, tools can be connected in chains in such a manner that each tool represents a certain step in the light propagation. This concept is easy to realize using the possibility of piping the output of one process to the input of another. Such a possibility exists under Unix, MSDOS and OS2. The approach also works under MacOS, provided that shells such as MPW or ToolServer are present. As far as each software tool represents a single step in light propagation, it is very easy to add special tools and to modify the existing set when a special problem arises. The currently existing set of tools includes models of coherent propagation, phase and intensity screens, the possibility of coherent mixing several light beams, input and output tools providing the possibility to import experimentally measured intensity and phase distributions along with presentation quality graphics output in many formats. A set of diagnostic tools provides information about the current parameters of the model.

Any Unix shell – see for example[1] or even script language such as Perl can be used to run LightPipes programs. The syntax of the Unix version of the package is described in detail in.[2]

2. COHERENT PROPAGATION MODELS

Models of light diffraction form the basis of any free-space physical optics package. The descriptions of the numerical approaches used in LightPipes follow.

2.1. Spectral algorithm

Let us consider the wave function U in two planes: $U(x,y,0)$ and $U(x,y,z)$. Suppose then that $U(x,y,z)$ is the result of propagation of $U(x,y,0)$ to the distance z, with the Fourier transforms of these two (initial and propagated) wave functions given by $A(\alpha,\beta,0)$ and $A(\alpha,\beta,z)$ correspondently. In the Fresnel approximation, the Fourier transform of the diffracted wave function is related to the Fourier transform of the initial function via the frequency transfer characteristic of the free space $H(\alpha,\beta,z)$, given by[34]:

$$H = \frac{A(\alpha,\beta,z)}{A(\alpha,\beta,0)} = \exp(-ikz(1-\alpha^2-\beta^2)^{1/2}) \tag{1}$$

where

$$A(\alpha,\beta,0) = \int\int_{-\infty}^{\infty} U(x,y,0)\exp(-ik(\alpha x+\beta y)\mathrm{d}x\mathrm{d}y \tag{2}$$

$$A(\alpha,\beta,z) = \int\int_{-\infty}^{\infty} U(x,y,z)\exp(-ik(\alpha x+\beta y)\mathrm{d}x\mathrm{d}y \tag{3}$$

Expressions (1, 2, 3) provide a symmetrical relation between the initial and diffracted wave functions in the Fresnel approximation. Applied in the order (2) $\Rightarrow$ (1) $\Rightarrow$ (3) they result in the diffracted wave function, while being applied in the reversed order they allow for reconstruction of the initial wave function from the result of diffraction. We shall denote the forward and the reversed propagation operations defined by expressions (1, 2 and 3) with operators L^+ and L^- respectively.

The described algorithm can be implemented numerically using the Fast Fourier Transform (FFT)[45] on a finite rectangular grid with periodic boundary conditions. This results in a model of beam propagation inside a rectangular waveguide with reflective walls. To approximate a free-space propagation, wide empty guard bands have to be formed around the wave function defined on a grid. To eliminate the influence of the finite rectangular data window, Gaussian amplitude windowing in the frequency domain should be applied – see[45] for extensive analysis of these computational aspects.

The simplest and fastest **LightPipes** command for propagation is **forvard**. It implements the spectral method described by (1, 2, 3). The syntax is simple, for example if you want to filter a field through a 1cm circular aperture and then propagate the beam 1m forward, you type:

```
begin 0.02 1e-6 |circ_ap 0.005 |forvard 1 > foo
file_int in < foo |cros_out out |file_ps out5.ps 128 >/dev/null
rm foo
```

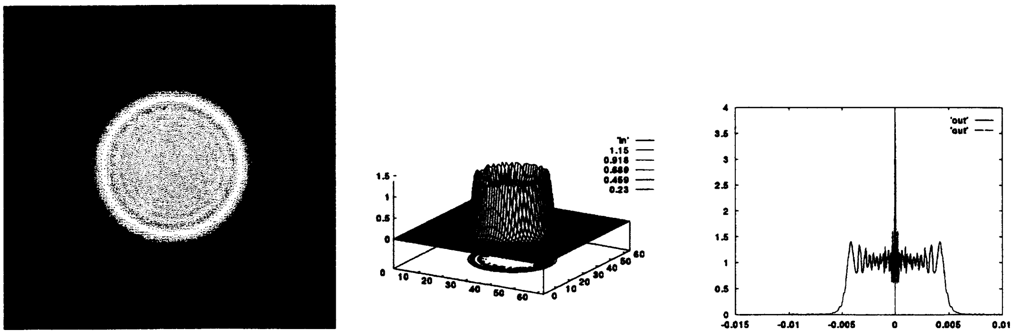


Figure 1. The result of the propagation: density, surface and cross section intensity plots

We see the diffraction effects, the intensity distribution is not uniform anymore.

The algorithm is very fast in comparison with direct calculation of diffraction integrals. Features to be taken into account:

- The algorithm realizes a model of light beam propagation inside a square waveguide with reflecting walls positioned at the grid edges. To approximate a free space propagation, the intensity near the walls must be negligible small. Thus the grid edges must be far enough from the propagating beam. Neglecting these conditions will cause interference of the propagating beam with waves reflected from the waveguide walls.
- As a consequence of the previous feature, we must be extremely careful when we propagate the plane wave to a distance comparable with D^2/λ, where D is the diameter (or a characteristic size) of the beam, and λ is the wavelength. To propagate the beam to the far field (or just far enough) we have to choose the size of our grid much larger than the beam itself, in other words we define the field in a grid filled mainly with zeros. The grid must be even larger when the beam is aberrated – divergent beams reach the region border sooner .

Spectral FFT algorithm has its drawbacks. The positive message is that it is very fast, works pretty well if properly used, is simple in implementation and does not require any allocation of extra memory. A negative argument may be supplied to `forward`. It means that the program will perform "propagation back" or in other words it will reconstruct the initial field from the one diffracted.

2.2. Direct integration as a convolution: FFT approach

Another possibility of a fast computer implementation of the operator L^+ is free from many of the drawbacks of the described spectral algorithm. The operator L^+ may be numerically implemented with direct summation of the Fresnel-Kirchoff diffraction integral:

$$U(x_1, y_1, z) = \frac{k}{2\pi i z}\int\int U(x, y, 0)\exp\{ik\frac{(x-x_1)^2+(y-y_1)^2}{2z}\}\mathrm{d}x\mathrm{d}y \tag{4}$$

with functions $U(x, y, 0)$ and $U(x, y, z)$ defined on rectangular grids. This integral may be converted into a convolution form which can be efficiently computed using FFT[6]CHulman1.[7] This method is free from many drawbacks of the spectral method given by the sequence (2) $\Rightarrow$ (1) $\Rightarrow$ (3), although it is still very fast due to its use of FFT for computing of the integral sums.

We'll explain this using a two-dimensional example, following,[6] p.100. Let the integral be defined in a finite interval $-L/2 \dots L/2$:

$$U(x_1, z) = \sqrt{\frac{k}{2\pi i z}}\int_{-L/2}^{L/2} U(x, 0)\exp\{ik\frac{(x-x_1)^2}{2z}\}\mathrm{d}x$$

Replacing functions $U(x)$ and $U(x_1)$ with step functions U_j and U_m, defined in the sampling points of the grid with $j = 0 \dots N$, and $m = 0 \dots N$ we convert the integral 5 to the form:

$$U_m = \sqrt{\frac{k}{2\pi i z}}(\sum_{j=1}^{N-1} U_j \int_{x_j-0.5}^{x_j+0.5} \exp\{ik\frac{(x_m-x)^2}{2z}\}\mathrm{d}x$$
$$+U_0 \int_{x_0}^{x_{0.5}} \exp\{ik\frac{(x_m-x)^2}{2z}\}\mathrm{d}x + U_N \int_{x_{N-0.5}}^{x_N} \exp\{ik\frac{(x_m-x)^2}{2z}\}\mathrm{d}x) \tag{5}$$

Taking integrals in 5 we obtain:

$$U_m = \sum_{j=1}^{N-1} U_j K_{mj} + U_0 K_{m0} + U_N K_{mN} \tag{6}$$

where: K_{m0}, K_{mj}, K_{mN} are analytically expressed with the help of Fresnel integrals, depending only on the difference of indices. The sums $\sum_{j=1}^{N-1} U_j K_{mj}$ can easily be calculated for all indices m as one convolution with the help of FFT.

The filter `Fresnel`, implements this algorithm for two-dimensional diffraction integrals. It is 2 to 5 times slower than spectral algorithm, it uses **8 times more memory** and it provides a "more honest" model of Fresnel diffraction. As it does not require any protection bands at the region boundaries, the model may be built in a smaller grid, therefore the resources consumed and time of execution are comparable or even better than that of **`forvard`**. **`Fresnel`** does not accept a negative propagation distance.

`Fresnel` does not produce valid results if the distance of propagation is comparable with (or less than) the characteristic size of the aperture, at which the field is diffracted. In this case **`forvard`** or **`steps`** should be used.

2.3. Finite difference method.

It can be shown that the propagation of the field U in a medium with complex refractive coefficient A, is described by the differential equation:

$$\frac{\partial^2 U}{\partial x^2} + \frac{\partial^2 U}{\partial y^2} + 2ik\frac{dU}{dz} + A(x,y,z)U = 0 \tag{7}$$

To solve this equation, we re-write it as a system of finite difference equations:

$$\frac{U^{k+1}_{i+1,j} - 2U^{k+1}_{i,j} + U^{k+1}_{i-1,j}}{(\Delta X)^2} + \frac{U^{k}_{i,j+1} - 2U^{k}_{i,j} + U^{k}_{i,j-1}}{(\Delta Y)^2} + 2ik\frac{U^{k+1}_{i,j} - 2U^{k}_{i,j}}{\Delta z} + A^{k+1}_{i,j}U^{k+1}_{i,j} = 0 \tag{8}$$

Collecting terms we obtain the standard three-diagonal system of linear equations, the solution of it describes the complex amplitude of the field in the layer $Z + \Delta Z$ as a function of the field defined in the layer Z:

$$-a_i U^{k+1}_{i-1,j} + c_i U^{k+1}_{i,j} - b_i U^{k+1}_{i+1,j} = f_i \tag{9}$$

where (we put $\Delta X = \Delta Y = \Delta$)

$$a_i = b_i = -\frac{1}{\Delta^2} \tag{10}$$

$$c_i = A^{k+1}_{i,j} - \frac{2}{\Delta^2} + \frac{2ik}{\Delta Z} \tag{11}$$

$$f_i = \frac{2ik}{\Delta Z}U^{k}_{i,j} - \frac{U^{k}_{i,j+1} - 2U^{k}_{i,j} + U^{k}_{i,j-1}}{\Delta^2} \tag{12}$$

The three-diagonal system of linear equations (9) is solved by the standard elimination (double sweep) method, described for example in.[9] This scheme is absolutely stable (this variant is explicit with respect to the index i and implicit with respect to the index j). One step of propagation is divided into two sub-steps: the first sub-step applies the described procedure to all the rows of the matrix, the second sub-step changes the direction of elimination and the procedure is applied to all the columns of the matrix.

The main advantage of this approach is the possibility to take into account uniformly diffraction, absorption (amplification) and refraction. For example, the model of a waveguide with complex three-dimensional distributions of the refraction index and absorption coefficient (both are defined as real and imaginary components of the (three-dimensional in general) matrix $A^k_{i,j}$) can be built easily.

It works also much faster than all the algorithms described previously on one step of propagation, though to obtain a good result at a considerable distance, many steps should be done. As the scheme is absolutely stable (at least for free-space propagation), there is no stability limitation on the step size in the direction Z. Large steps cause high-frequency errors, therefore the number of steps should be determined by trial (increase the number of steps in a probe model till the result stabilizes), especially for strong variations of refraction and absorption inside the propagation path.

Zero amplitude boundary conditions are commonly used for the described system. This, again, creates the problem of wave reflection at the grid boundary. The influence of these reflections can, in many cases, be reduced by introducing an additional absorbing layer in the proximity of the boundary, with the absorption increasing smoothly (to reduce the reflection at the absorption gradient) towards the boundary.

The finite difference algorithm is implemented in a filter **steps**. **Steps** accepts 6 arguments, first - the step size, second- the number of steps, third and fourth - files containing absorption and refraction coefficients to be imported and the last two arguments are the filename where field cross-sections are saved after each N steps, where N is the last argument. All filenames are optional.

For example:

```
begin 0.004 0.63e-6  | circ_ap 0.0015 | lens 0.5 |\
steps 0.01 75 void void steps_out > /dev/null
```

produces the file steps_out which can be used to plot the radial intensity distribution in the propagating beam.

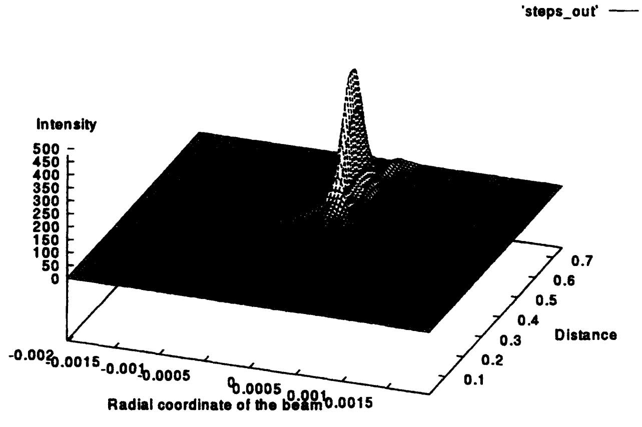


Figure 2. Steps output for a simple lens.

Steps has a built-in absorption layer along the grid boundaries (to prevent reflections), occupying 10% of grid from each side. Steps can import absorption/refraction coefficients defined in the format similar to the one exported by **file_int**. Steps is the only filter in LightPipes which allows modeling of (three-dimensional) waveguide devices. Like **forvard**, Steps can inversely propagate the field, for example the sequence ...|**steps 0.1 1 | steps -0.1 1 |...** doesn't change the field distribution.

3. EXAMPLE MODELS

3.1. Twyman-Green interferometer

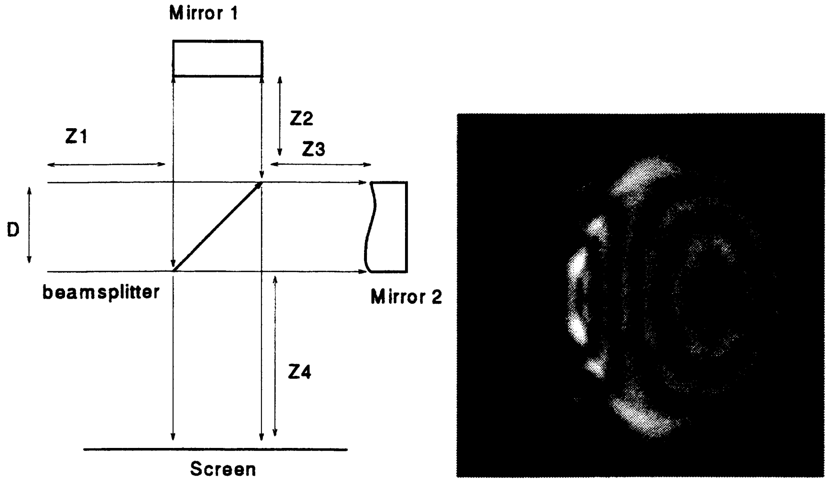


Figure 3. Twyman-Green interferometer (left) and the output of the interferometer model

The scheme of the interferometer is shown in Fig. 3. In this example we put D = 1cm, $Z_1 = 0.5m$, $Z_2 = 0.2m$, $Z_3 = 0.4m$, $Z_4 = 1m$, $\lambda = 0.5\mu m$. The first mirror is plane and the second is aberrated (coma) with aberration amplitude of $1\mu m$. The beam-splitter is ideal dividing the beam 3:7. We run the following script:

```
#!/bin/csh -f
# propagation and splitting
begin 0.03 5e-7 | circ_ap 0.005 | forvard 0.5 | b_split foo 0.3 >foo1
# Propagation to mirror1, reflection,
# propagation back and beam-splitter again
forvard 1 <foo|i_filter 0.7 >foo2
# Propagation to mirror2, reflection,
# propagation back and beam-splitter again
forvard 0.4 < foo1 | Zernike 3 1 0.005 25|forvard 0.4|i_filter 0.3 >foo3
# mixing the two beams and propagation to the screen
b_mix foo2 <foo3 |forvard 1| interpol 0.012 128 | file_ps tg.ps 128>/dev/null
# deleting the temp files
rm foo foo1 foo2 foo3
```

3.2. Unstable laser resonator

This example of calculation of the output mode of an unstable resonator with square mirrors is taken from.[8] The scheme of the resonator is shown in Fig. 4 together with the equivalent lens waveguide scheme used in the calculations. An infinite sequence of reflections from resonator mirrors is replaced here by finite but long enough sequence of propagations through afocal magnifying lens system.

```
#!/bin/csh -f

# LightPipes script: a model of an unstable resonator
# with N=10, M=2,
```

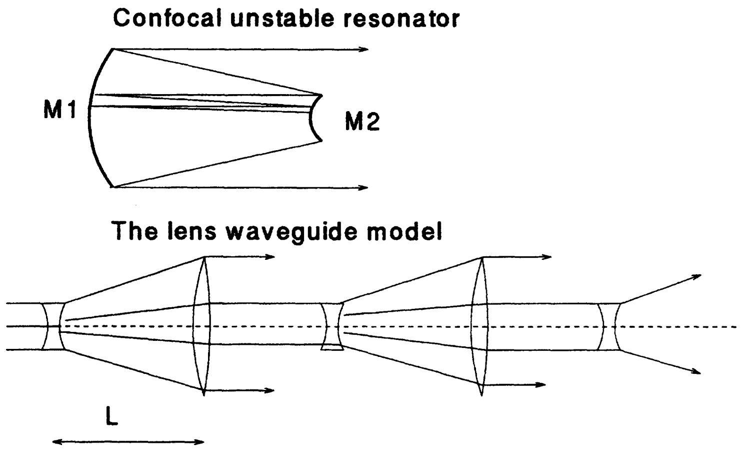


Figure 4. An unstable confocal resonator as a lens waveguide

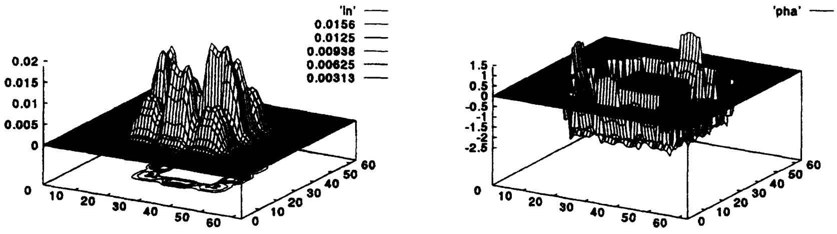


Figure 5. Intensity and phase distributions at the output of the model of unstable resonator

```
# See D.B. Rench, Applied  Optics 13, 2546...2561 (1974)
# All dimensions are taken from the table on p. 2552
# in this reference.
# Theoretically the initial field distribution may be random
# but we'll use a good plane wave because it won't
# converge fast with a random distribution
begin 20. 3e-4 512 >field1
# 20 field iteration inside the resonator
@ i=0
while( $i < 20)
# reflection from the convex mirror and propagation
rect_ap 5.48 < field1 | lens -1e4 |forward 1e4 > field2
# reflection from the concave mirror and propagation
rect_ap 10.96 <field2 | lens 2e-4  | forward 1e4 | normal y > field1
Strehl y < field1 >/dev/null
@ i += 1
end
# output of the resonator is screened by the output mirror:
# the output
rect_screen 5.48 <field1 | file_int in |\
 file_pha pha|file_ps res_out.ps |Strehl y>/dev/null
```

```
# removing the temp files
rm field1 field2
```

When executed, the script prints values of the feedback coefficient and other information at each iteration. From the output sequence we see that after 20 iterations the feedback coefficient stabilizes, so the calculations can be terminated. If the files `field1` and `field2` are not deleted, we can continue the calculations later. To do this, the operator **begin** should be commented, othervise the calculations will be started from a plane wave.

In this example we calculated the mode profile of an ideal unstable resonator. A resonator with aberrated and misaligned mirrors can be also easily modeled. One problem with the considered model is the way, the mirror's curvature is defined. Curved mirrors are modeled as phase masks `lens` and for high curvatures the phase difference between the neighbour points of grid may be higer than $\pi/4$, especially in the proximity of the mirror edges. If the phase difference is greater than $\pi/4$, the results are not valid, so the nuber of points must be increased to fix the mentioned phase condition. It leads to enormous grids (we used 512x512) for a simple problem. This can be solved by applying spherical coordinate system to propagate spherical waves. In such a system the coordinate surfaces coincide approximately with the propagating wavefront. The script for a laser with a confocal unstable resonator in spherical coordinates follows (including laser medium):

```
#!/bin/csh -f
#all dimensions are in mm here
begin 7. 3e-4 100 >field1
# 40 iterations inside the resonator
@ i=0
while( $i < 20)
# reflection from the convex mirror and propagation
rect_ap 5.48 < field1|l_amplif 1e-4 1e4 1 | lens_fresn -1e4  1e4 > field2
# reflection from the concave mirror and propagation
rect_ap 10.96 <field2 |l_amplif 1e-4 1e4 1| lens_fresn 2e4  1e4\
|Strehl y > field1
cp field1 field_out
interpol 7. <field1 > field2
mv field2 field1
@ i += 1
end
# output of the resonator is screened by the output mirror:
# the output
echo 'Output beam:'
cat field_out |convert y |rect_screen 5.48 | file_ps res_out.ps\
 |Strehl y >/dev/null
# removing the temp files
rm field1 field2
```

Here is the tail of the screen output:

```
Output beam:
Strehl: ratio= 8.209055e-01 energy= 2.917869e+02
Center_of_gravity: x= -1.716706e-03 y= -1.600309e-03
Grid size: 1.400000e+01, Grid sampling: 100
```

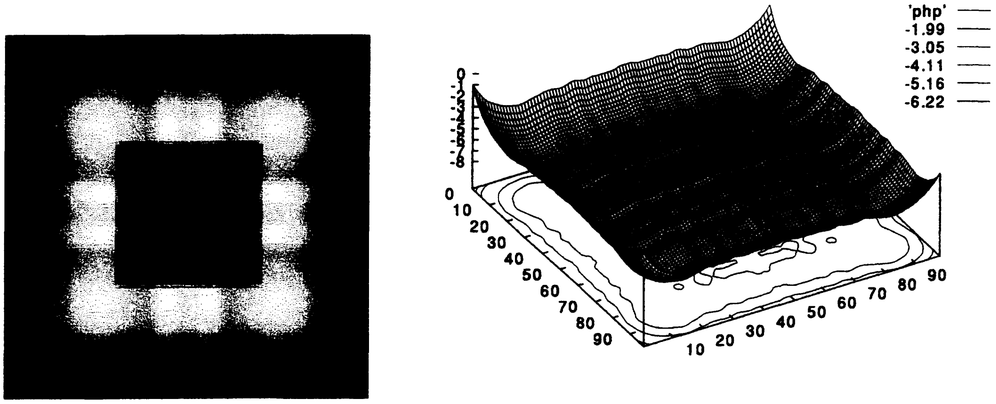


Figure 6. The output (intensity and phase) of an unstable resonator, modeled in spherical coordinates

The last script is much faster and the field occupies a larger part of the grid. It produces almost the same feedback coefficient, the Strehl number is also close to that obtained in the previous example, the difference is due to the better computational approach used in the latter case.

4. LIGHTPIPES PORTS

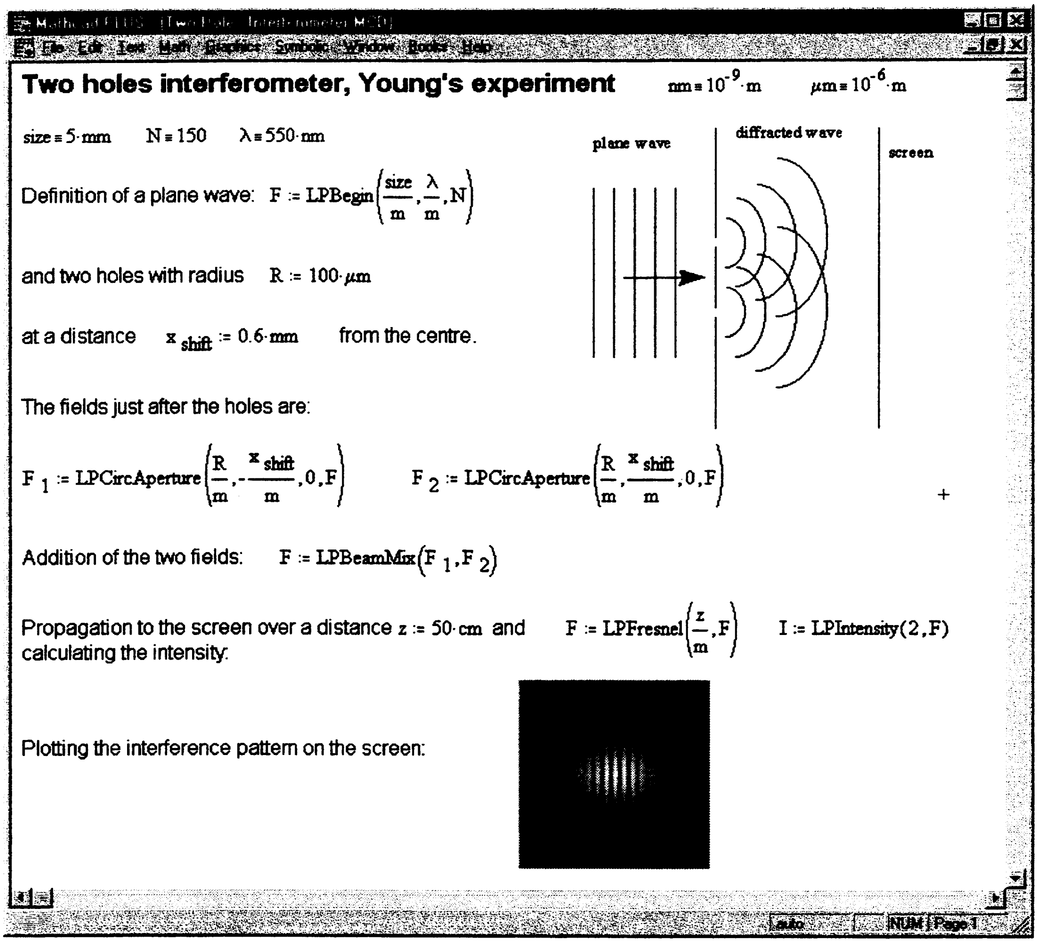
4.1. LightPipes for Mathcad.

Mathcad from MathSoft Inc. is known to be one of the most user-friendly environments for handling mathematics, text, graphics, animations and programs combined in one single document. Besides a large number of functions internal to Mathcad, the professional versions of Mathcad offer the oppertunity to add compiled libraries of user-written functions. The translation of the LightPipes tools into a Dynamic Linked Library (DLL) for Mathcad resulted in an optical toolbox with high execution speed and a high level of flexibility and ease of use for the user. Figure 7 shows a simple application of LightPipes for Mathcad.

The Lightpipes commands can be very well commented with text and figures, and the results are easily presented using Mathcad's graphical and animation features. Furthermore the document can easily be modified and executed within the program without involving text editors or compilers. The program feature of Mathcad makes it possible to perform iterations which can be used to demonstrate laser resonators or phase retrieval. These and several other examples, including an on-line manual of LightPipes for Mathcad can be found in reference .[10]

4.2. LightPipes on the Internet.

The internet has great potential for computer assisted education and several attempts have been made to build courses which can be done by the student on her of his personal computer. Using the platform independent programming language 'Java' so-called 'applets' can be written which can be embedded in HTML-pages to be displayed with a web-browser like Internet Explorer and Netscape. As the Java compiler does not generate processor code but an intermediate 'byte-code' for a virtual machine a second stage is necessary to run the program on the user's computer. Because of this the execution of Java-applets is slow (5-10x slower) compared to compiled C-programs. All though modern browsers have a so-called just-in-time compiler to increase speed, high performance applications having extensive codes will enhance the time for loading the applet requiring relative high-speed internet connections. Fortunately, Netscape introduced the possibility to produce plug-ins which are pre-compiled (C-) routines which must be down-loaded by the user once, extending the capabilities of the browser.[11] Plugins can be made for all important platforms.



Two holes interferometer, Young's experiment $nm \equiv 10^{-9} \cdot m$ $\mu m \equiv 10^{-6} \cdot m$

$size \equiv 5 \cdot mm$ $N \equiv 150$ $\lambda \equiv 550 \cdot nm$

Definition of a plane wave: $F := LPBegin\left(\frac{size}{m}, \frac{\lambda}{m}, N\right)$

and two holes with radius $R := 100 \cdot \mu m$

at a distance $x_{shift} := 0.6 \cdot mm$ from the centre.

The fields just after the holes are:

$F_1 := LPCircAperture\left(\frac{R}{m}, -\frac{x_{shift}}{m}, 0, F\right)$ $F_2 := LPCircAperture\left(\frac{R}{m}, \frac{x_{shift}}{m}, 0, F\right)$

Addition of the two fields: $F := LPBeamMix(F_1, F_2)$

Propagation to the screen over a distance $z := 50 \cdot cm$ and calculating the intensity: $F := LPFresnel\left(\frac{z}{m}, F\right)$ $I := LPIntensity(2, F)$

Plotting the interference pattern on the screen:

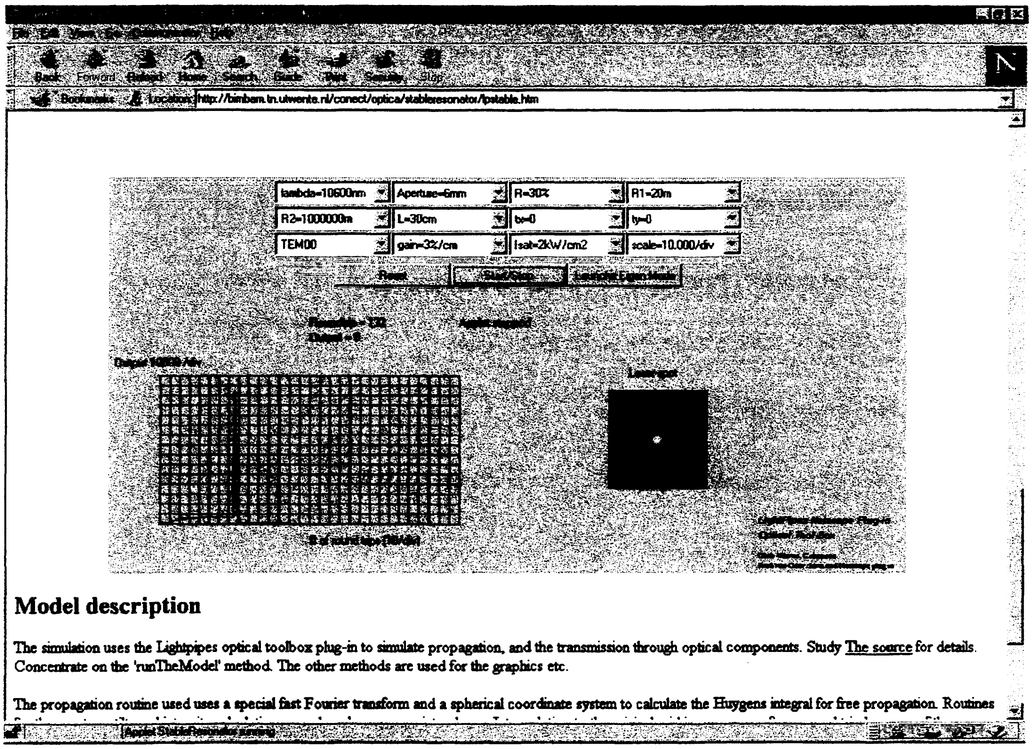
Figure 7. A two-holes interferometer modeled in LightPipes for Mathcad.

Because LightPipes demands as much as computer power as possible we made a Netscape-Plugin which contains a selection of the LightPipes routines. After placing the DLL in the Netscape Plugin directory the user can load HTML pages containing small Java applets which call the LightPipes routines from the plug-in DLL. Loading the applet can be done in short time because besides the LightPipes routines-calls, the applet only contains the commands for communication with the user and for presenting the (graphic al) results in the applet-window. In this way, inter-active optical simulations can be executed within reasonable time exploring the maximum available computer power. Figure 8 shows an example of the simulation of a laser with a stable resonator including saturable gain.

The user can alter parameters like mirror curvature, aperture diameter, etc. during the execution and observe the effect on the transversal mode structure and the output intensity of the laser. More examples of simulations with LightPipes on the internet can be found on our web-site.[10]

4.3. PipesSA a portable C++ version of LightPipes

The C^{++} port of the LightPipes package, PipesSA, was constructed because the pipe based version worked very slowly on a Macintosh. This is caused by the relatively slow disk input/output, at least on the machine of one of the authors.



http://bimbam.tn.utwente.nl/conect/optica/stableresonator/lpstable.htm

lambda=10600nm Aperture=6mm R=30% R1=20m
R2=1000000m L=30cm tx=0 ty=0
TEM00 gain=3%/cm Isat=2kW/cm2 scale=10.000/div

Model description

The simulation uses the Lightpipes optical toolbox plug-in to simulate propagation, and the transmission through optical components. Study The source for details. Concentrate on the 'runTheModel' method. The other methods are used for the graphics etc.

The propagation routine used uses a special fast Fourier transform and a spherical coordinate system to calculate the Huygens integral for free propagation. Routines

Figure 8. Simulation of a stable laser resonator with LightPipes using a Plug-in within a HTML page. After 75 round trips the user decreased the out coupler reflectivity from 99% to 30% to study 'Q-switching' of the laser.

The idea was to create a stand alone version of the tools, that otherwise only work within the MPW shell (a Macintosh developers environment), with a datastructure in it that holds the intermediate light beams. This means that the piping mechanism is no longer possible but as a compromise a command parser was included in the program. Each filter operates on this data structure and no input/output is required. Each command line should only contain a single filtering command.

The lightpipes program should be inserted in an ascii file that can be loaded into PipesSA or entered at the **'Pipes>'** promt directly.

The ascii file should read:

```
begin 0.1 1e-6 256
circ_ap 0.025
file_ps ps0
fresnel 100
file_ps ps1
quit
```

As stated, these commands can be entered directly, or the file can be loaded using the 'load filename' command. If the file with 'filename' exists, the file is read and the commands are executed sequentialy.

Other commands that are now included in PipesSA are obtained by starting up the program and entering the 'help' command. The result is given in the following output of PipesSA.

```
Pipes> help
The implemented commands so far are:

begin      circ_ap    rect_ap    circ_sc    rect_sc
gauss      gauss_sc   zernike    lens       tilt
file_ps    file_pgm   file_int   file_pha   cros_out
b_split    b_mix      read       write      load
forvard    fresnel    steps      forward    pip_fft
unwrap     interpol   interp1    strehl     random
gamma      normal     absorber   amplifier

!!other commands are soon to follow!!

Pipes> quit
```

To get information on these commands the command 'help command_name' can be used, but not all commands have been given this option yet. For a full discription on all commands the Unix manual can be used.

This command driven program has been written in standard C^{++} to enable the porting to other operating systems. It was found that this standalone version performs faster than the pipe based version on both HP and MSDOS machines.

REFERENCES

1. G. Anderson, P. Anderson, The Unix C shell guide, Prentice-Hall (1986)
2. LightPipes: beam propagation toolbox, G. Vdovin, OKO Technologies, 1997 (also available electronically at http://xs4all.nl/~oko/pipes/).
3. J.W. Goodman, Introduction to Fourier Optics, McGraw–Hill, 49–56 (1968)
4. W.H. Southwell, J. Opt. Soc. Am., **71**, 7–14 (1981)
5. A.E. Siegman , E.A. Sziklas, Mode calculations in unstable resonators with flowing saturable gain: 2. Fast Fourier transform method, Applied Optics **14**, 1874-1889 (1975)
6. N.N. Elkin, A.P. Napartovich, in "Applied optics of Lasers", Moscow TsniiAtomInform, 66 (1989) (in Russian)
7. R. Baracat, in "The Computer in Optical Research", ed. B.R. Frieden, Springer-Verlag, 72–75 (1980)
8. D.B. Rench, Three- dimensional Unstable Resonator Calculations with Laser Medium, Appl. Opt. **13**, 2546-2561 (1974)
9. A.A. Samarskii, E.S. Nikolaev, Numerical Methods for Grid Equations, V.1, Direct methods, pp. 61–65, Birkhäuser Verlag 1989.
10. F.A. van Goor, LightPipes for Mathcad and Java. On web-site: `http://bimbam.tn.utwente.nl/conect/optica/`
11. S. Narayanan, 'Netscape Client Plug-Ins'. A step-by-step procedure to deve lop a Netscape Plug-in using MS Visual C++ 4.0. On web site: `http://developer.netscape.com/news/viewsource/naru_plugins.html`