Lab 1_Computation of diffraction patterns

This lab is concerned with the methods of calculating diffraction patterns by using Huygens' Principle (Details please see the web pages as follows: http://www.mathpages.com/home/kmath242/kmath242.htm and http://en.wikipedia.org/wiki/Huygens%E2%80%93Fresnel_principle). It is assumed that you have known MATLAB and can understand how it works with matrices.

When light travels past an object or through an opening, it spreads out or diffracts. The pattern of light intensity this produces by using Huygens' Principle can be calculated. This states that every point on a wavefront can be considered as a source of secondary wavelets which spread out in all directions. To find the resultant displacement at any point, all the individual displacements produced by these secondary waves can be combined, using the superposition principle and taking into account their amplitudes and relative phases.

Consider a plane wave impinging on an attenuating wall which has a tiny hole on it. The hole acts as a wave source, **S**, can generate a series of spherical waves, which we can detect when the wave propagate and reach a screen ahead with a given distance.

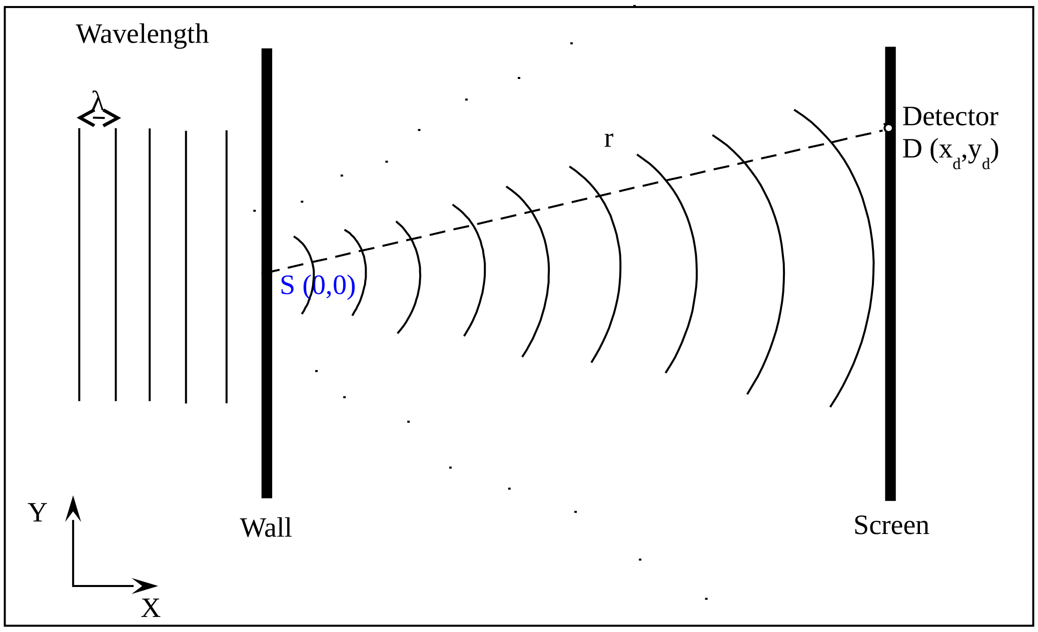

Fig. 1. Schematic of diffraction

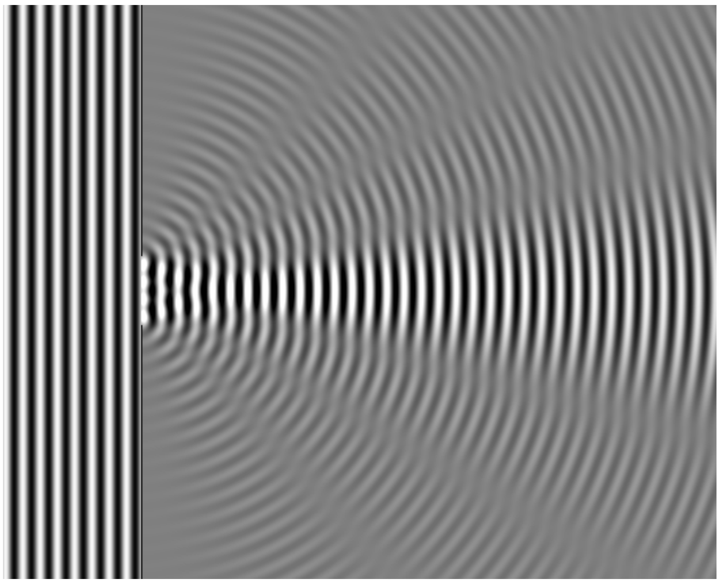

Fig. 2. Numerical simulation of diffraction pattern with an incident plane wave (Simulated by 1-D beam propagation method).

Now imagine that the electric field variable close to the point **S**, which we write ***E_s***, is varying sinusoidally with time, with an angular frequency ***ω*** and amplitude ***E_max***,

$$E_s(t) = E_{\max}\cos(\omega t) \tag{1.1}$$

Then the field at the point **D**, a distance **r** away, will also be varying sinusoidally. The intensity of the light at that point falls off as ***1/r***, because it spreads out uniformly in all directions. Therefore its amplitude can be written as ***A/r***, where ***A*** is a constant. Its phase will depend on how many wavelengths fit into the distance ***r***.

$$E_d(t) = \frac{A}{r}\cos(\omega t - {}^{2\pi r}\!/_{\lambda}) \qquad \text{where } r = \sqrt{x_d^2 + y_d^2} \tag{1.2}$$

the intensity of the light at **D** is equal to the time average value of E_d^2, multiplied by $\varepsilon_0 c$. c is the propagating speed of wave, and ε_0 is the medium constant.

$$\text{Intensity at D} = \varepsilon_0 c\left\langle E_d(t)^2\right\rangle = \frac{\varepsilon_0 c}{2}\times\frac{A^2}{r^2} \tag{1.3}$$

Note, if the amplitude of the oscillation is unknown, such as **A**, in principle, the field at time **t = 0** and a quarter of a period later (**t = τ/4 = π/2ω**) can be measured. Then,

$$E_d(0) = \frac{A}{r}\cos(-{}^{2\pi r}\!/_{\lambda}) = \frac{A}{r}\cos({}^{2\pi r}\!/_{\lambda}) \tag{1.4}$$

$$E_d({}^{\tau}\!/_{4}) = \frac{A}{r}\cos({}^{\pi}\!/_{2} - {}^{2\pi r}\!/_{\lambda}) = \frac{A}{r}\sin({}^{2\pi r}\!/_{\lambda}) \tag{1.5}$$

So if you square and add the two measurements, the fluctuations due to the sinusoidal variation disappear (because **$\sin^2 + \cos^2 = 1$**), and the intensity at **D** can be given by

$$\text{Intensity} = \frac{\varepsilon_0 c}{2}(E_d(0)^2 + E_d(\tau/4)^2) \qquad (1.6)$$

Now consider that the source consists of a number, **M**, of similar point sources.

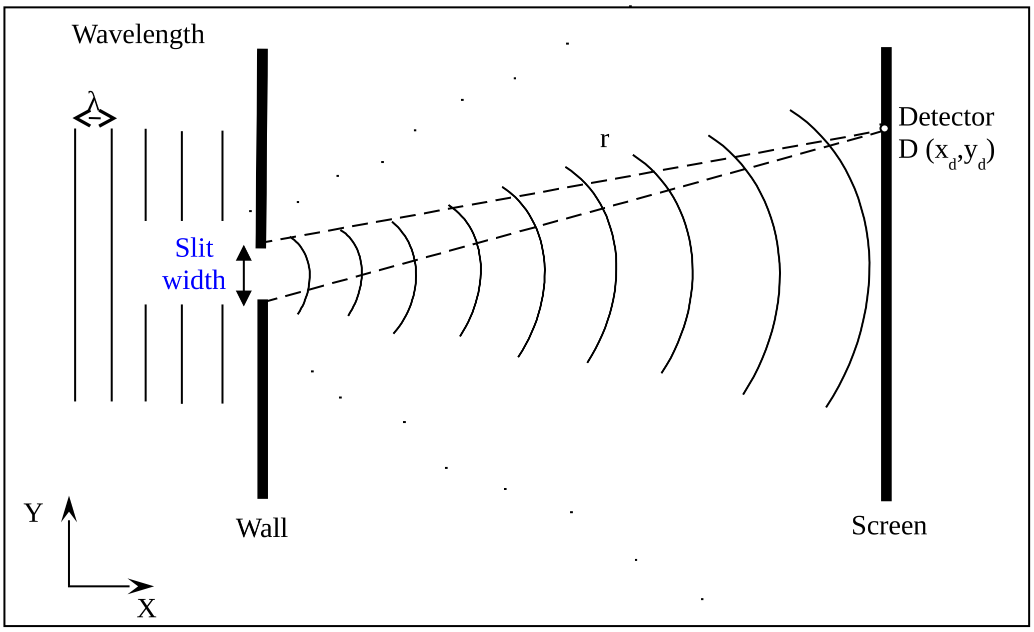

Fig. 3. Schematic of diffraction with slit width

If the sources are located at the points, such as (***x***$_{sj}$***, y***$_{sj}$), each with the same initial amplitude **(*A*)**, the total field at the point **D**, E_{total}, is given by

$$E_{total}(0) = \sum_{j=1}^{M} \frac{A}{r_j}\cos(2\pi r_j/\lambda) \text{ and } E_{total}(\tau/4) = \sum_{j=1}^{M} \frac{A}{r_j}\sin(2\pi r_j/\lambda) \qquad (1.7)$$

$$\text{Where } r_j = \sqrt{(x_d - x_{sj})^2 + (y_d - y_{sj})^2} \qquad (1.8)$$

And the total intensity can be expressed as:

$$I_{totoal}(x_d, y_d) = \frac{\varepsilon_0 c}{2}(E_{total}(0)^2 + E_{total}(\tau/4)^2) \qquad (1.9)$$

It would simplify the equation (1.9) if we could ignore the constant quantity $\varepsilon_0 c$. In all the computations, the amplitude of the wave at the source is arbitrary. Thus we will never calculate the absolute values of intensity. So we can choose the way intensity is measured so that this constant is effectively equal to 1 (Normalization process). In such a measurement system,

$$I_{totoal}(x_d, y_d) = \frac{1}{2}(E_{total}(0)^2 + E_{total}(\tau/4)^2) \qquad (1.10)$$

We keep the factor **1/2** because it has a relevant meaning. It states that the intensity is related to the time average of the amplitude squared. The intensity pattern produced by light coming through an opening (or openings) is quite complicated in the immediate vicinity of the opening (the Fresnel regime) but much simpler if you go far away (the Fraunhofer regime). In our labs, you will investigate where the transition between these two regimes is located, and how its location depends on the size of the opening and on the wavelength of the light.

CO_1.1 Visualization of wave fields

We start to investigate how the field from a point source varies as we move the detector to all points throughout the space between the wall and the screen. This involves writing a short script to calculate the field **E** at the points **D**, launching from a single point source at **S**, using the equation (1.2).

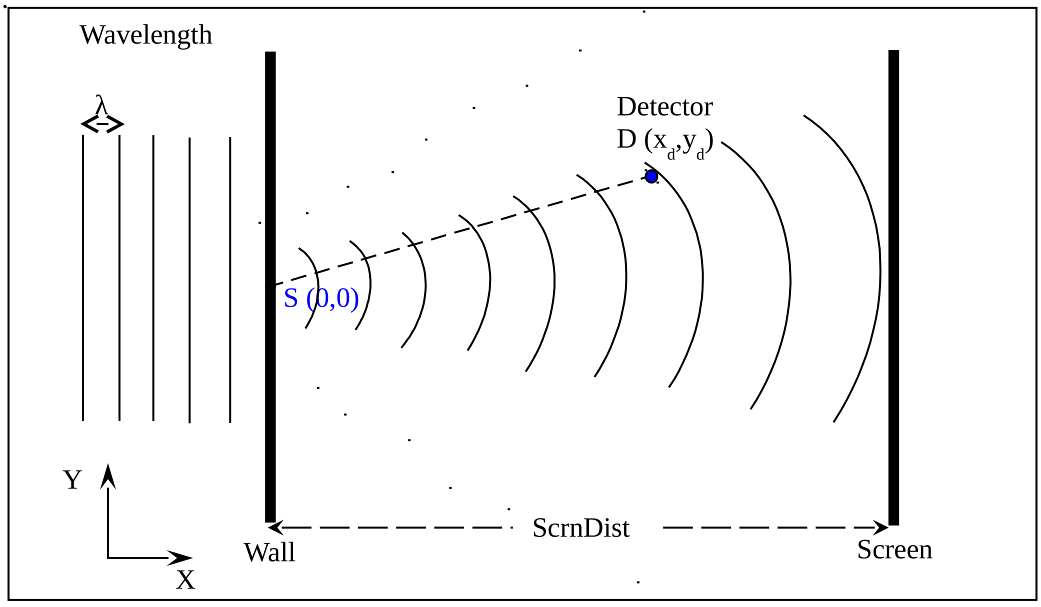

Fig. 4. Schematic of diffraction with detector

This script can be found on the web page for this module as an m-file called **CO_lab1a.m**. Download this file to your PC and open it. Note this script can be run on your current directory at any time, simply by typing its name in the command window.

```
>> CO_lab1a;
```

(You don't type the ">>" of course. That is the MATLAB prompt. And don't forget MATLAB is case sensitive.)

I. Firstly we must choose the variables to work with, and set them to appropriate values. For this exercise we will use the following (all lengths are expressed in a meters):

lambda = 4.0e-3; % wavelength: (4.0 mm)

scrnDist = 5.0e-2; % distance to the screen: (50 mm)

scrnWdth = 2.4e-2; % width of the screen: (±12 mm)

xs = 0; % x-coord of the source:

ys = 0; % y-coord of the source:

A = 1 (arbitrary); % amplitude of the source:

Notice that we have chosen an unusually large value for the wavelength. In the following labs we will treat the waves as the visible optical wavelength range, that is, wavelengths between 400–800 nm. However right now we want to see the significant oscillatory behaviour of the waves, and also we will pretend we are doing these labs with microwaves.

II. In order to calculate **E** throughout the space between the source and the screen. In the script we have chosen a rectangular array of detection points whose x-coordinates vary between close to **0** and **scrnDst** and whose y-coordinates vary between **-scrnWdth/2** and **+scrnWdth/2**.

III. Note in the script the two lines of code which calculate the field variable ***E*(0)**. As usual, if you don't know already, ask about the meaning of the "." before the "^" and the "/" (We have done corresponding exercises in the module of "Matlab tutorial"). It should be clear that the relationship between these commands and the basic equation we are working with the Equation (1.2). Check that the script has calculated one value of **E0** for each of the 6 × 5 detection points, and that those values look reasonable — the field should be large near the source and decrease further away from it.

IV. Change the size of the array of detection points, by setting the variable N equal to 500 rather than 5. You will need to make the change inside the script, and **save** it into the working directory, as **CO_lab1b.m**. Then run the script (by typing its name in the command window, or just press the button of **F5** on the keyboard). Investigate the values of the field variable *E* between the source and the screen by plotting it as a 3-D plot — xd and yd along the *x*- and *y*-axes, and **E0** along the *z*-axis. You do this by typing this statement, which is one of the standard MATLAB functions, in the command window:

```
>> mesh(xd,yd,E0);
```

Check that you understand what the plot is telling you. Change the value of wavelength inside the script to 8 mm and then to 2 mm. Save each time, run and re-plot. Are the changes in the figure as you expected?

(e) Another useful way to represent the field is to do a **contour plot in color**.

Close the mesh plot and type this statement in the command window:

```
>> ContourPlot(xd,yd,E0);
```

Check that you understand all the information in the plot. Try three different wavelengths, 8 mm, 4 mm and 2 mm. Save each time, run and re-plot. Is everything still as you expected?

CO_1.2 Point source interference

This lab is to calculate the field due to two point sources separated by a small distance.

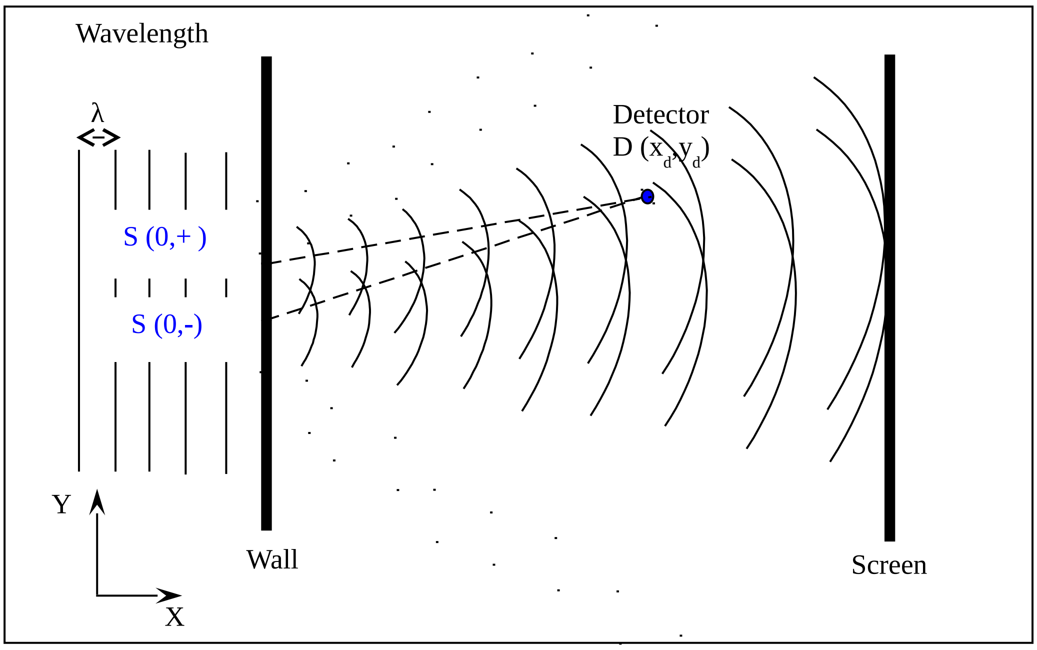

Fig. 5. Diffraction with interference

You need to extend the script used in problem 1, to get it to handle two sources instead of one. So, before you start, open that script and save it in the current directory under the name **CO_lab1c.m**. (Use “Save as” in the menu of “File”.) You now have a second copy in which can now make changes.

I. Firstly you need to add another parameter to the ones that are set at the top of your script—the separation between the two sources. Separation of sources:

   ```
   srceSepn = 1.2e-2; % (1.2 cm)
   ```

   Write the appropriate statement into the script directly under the other assignments.

II. There are two sources now, so we will make the **x**- and **y**-coordinates of the source a (1 × 2) row vector. The appropriate statements are:

   ```
   xs = [0 , 0];
   ys = [-srceSepn/2 , srceSepn/2];
   ```

   Replace the statements that originally set **xs** and **ys** to zero with these two statements.

III. Since there are two sources at the moment, we need to calculate two arrays of values of **r** between each source and all the detector points. Simply replacing the single statement that calculated **r** previously with these two statements,

   ```
   r1 = sqrt((xd-xs(1)).^2 + (yd-ys(1)).^2);
   r2 = sqrt((xd-xs(2)).^2 + (yd-ys(2)).^2);
   ```

   Then the field variable at **t = 0**, at any of the detector points is the sum of two terms,

   ```
   E0 = A*cos(2*pi*r1/lambda)./r1 + A*cos(2*pi*r2/lambda)./r2;
   ```

   Make these changes in the script, save and run it.

IV. Examine the results of the calculation by again plotting **E(0)** as a 3-D plot (mesh) and as a contour plot (ContourPlot). Describe briefly the main characteristics of the pattern you observe in an attached paper when you get a contour plot.

V. Now change the wavelength. Try three different wavelengths: 8 mm, 4 mm and 2 mm. Plot the results as a contour plot and describe in the attached paper how the contour plot changes as the wavelength increases.

VI. Set the wavelength back to 4 mm and change the separation between the two sources to 24 mm. Describe in the attached paper how the contour plot changes as the source separation increases.

CO_1.3 Interference of two points source

This next lab involves investigating the interference effects we have been observing by measuring the intensity of the wave field at points along the screen. Assuming there is a row of detectors attached to the screen as in the diagram in Fig. 1, rather than a rectangular array of detectors filling the area between sources and screen. Then these measurements can be plotted as a 1-dimensional graph of intensity versus the y-coordinates of the detectors.

I. Firstly we need to calculate the intensity using the equation (1.10). This involves calculating the wave field at time **$t = \tau/4$** as in the equation (1.7). We need to create a new variable to denote this quantity as **E4**. So these two statements need to added to the end of the script.

```
E4 = A*sin(2*pi*r1/lambda)./r1 + A*sin(2*pi*r2/lambda)./r2;
Itot = (E0.^2 + E4.^2)/2;
```

Make these changes and save the new script as **CO_lab1d.m**. However, if you try to observe I_{tot} by plotting it using **mesh** or **ContourPlot** you won't be able to detect much useful information. The relative intensities are too small to register clearly on the graph.

II. Run the script from Part (a) with these new parameters (optical wavelengths range now!),

wavelength = 600 nm screen distance = 50 mm

screen width = 30 mm source separation = 0.016 mm

The intensity only at points on the screen can be plotted. The coordinates of the detector points on the screen are [xd(:,N),yd(:,N)], and the field intensities at those points are Itot(:,N). Remember, if you don't know what ":,N" means, ask. And to plot these values of **Itot** against y on the screen you type, in the command window:

```
>> plot(yd(:,N),Itot(:,N));
```

III. Before investigating this pattern, save a copy of it for the future labs. Make sure the figure is active by clicking inside it. Choose "Save as" from the "File" menu. Instruct it to be saved as a MATLAB figure, and give it the name **CO_lab1_fig1.fig**.

Think & answer the following questions:

1. Why is the total intensity zero at some points along the screen?

2. Do the heights of the maximums change as you go away from the centre? Why?
3. How does the pattern change if you increase the wavelength of the light? Why?

IV. The theory of two-slit diffraction predicts that the maximums and minimums of these interference fringes should be arranged along the y-axis at distances from the centre of the screen given by,

$$y_m = m\lambda \times \frac{ScrnDist}{\mathrm{SrceSepn}} \quad \text{or} \quad y_m = \left(m + \frac{1}{2}\right)\lambda \times \frac{ScrnDist}{\mathrm{SrceSepn}} \qquad (1.11)$$

for a maximum (bright fringe) or a minimum (dark fringe), respectively, where **m** is an integer.

Measure the positions of several of these fringes, both bright and dark. Make sure your detector points along the screen are close enough together — **N** should be at least 1000. Calculate the equivalent value of m for each (using the equation of (1.11)) and write them down.

V. At last, there is another way of representing the intensity at points along the screen, and that is by using a grey scale representation. Download from the module's web pape, and open & run it by typing in the command window,

```
GS(yd(:,N), Itot(:,N));
```

This way of representing the data assumes that the sources in the figure 1 and 5 are not small round holes, but long narrow slits, parallel to the z-direction, perpendicular to the plane of the paper.

CO_1.4 Slit of finite width

This lab is to calculate the intensity due to a slit of finite width (by which we mean that the width is normally small but non-zero).

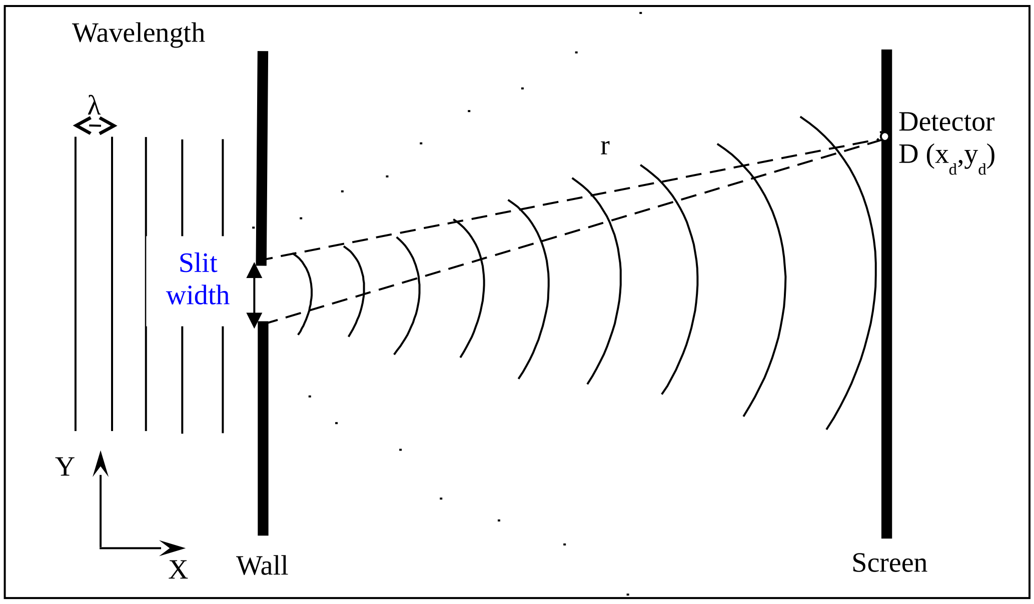

Fig. 6. Diffraction with a finite slit width

The computing task is quite straightforward. The slit through which light passes can be considered as equivalent to a row of point sources. The light reaching the screen from the slit is therefore the superposition of the light coming from all of the individual equivalent point sources. In lab CO_1.2&3 we calculated the superposition of light from two sources. What we will work with from now on is essentially an extension of that script, but with some important structural changes. You will find it in the webpage under the new name **Huygens.m**.

I. In order to investigate how light diffracts with optical wavelengths in the range 400~800 nm. To keep the diffraction patterns looking much the same as before we will also need to make the width of the slit smaller — such as the order of 10^{-9} meter. Therefore enter these statements into the command window (**remember all units of length are in meters**):

```
>> global lambda scrnDist scrnWdth slitWdth;
>> lambda = 600e-9;
>> scrnDist = 50e-3;
>> scrnWdth = 30e-3;
>> slitWdth = 4.0e-6;
```

In the script, one can see that the global statement repeated there. So these values are now accessible to the script without having to be set each time it is run. Note that the numbers of sources and detectors have been set to 7 and 5 respectively. This is for present convenience. We'll increase them later.

II. The other big structural change can be seen in the first and last lines. The script has been turned into a MATLAB function. When you want to do the calculations it contains you call it like this. **Caution: the global variables must still be set.**

```
>> [yd Itot] = Huygens;
```

and MATLAB will generate two (5×1) column vectors for **yd** and **Itot**. You can output the figure by the command of “**plot**”. A call like this can be made from inside another script, and the “return” at the end of the file tells MATLAB to go back and execute the next statement after the point at which the function was called.

III. Look at the rest of the file, and see how the calculation is done. The coordinates of the sources, **xs** and **ys**, are calculated as they were before, except that now they are row vectors and not just pairs of numbers. Likewise the coordinates of the detectors, **xd** and **yd**, are calculated similarly, as column vectors. There are **M** sources and **N** detectors; therefore there are $M \times N$ different distances between any one of the sources and any one of the detectors.

The program calculates all of these in one $M \times N$ matrix. To do this the program replaces each of **xs**, **ys**, **xd** and **yd** by matrices of this same size, the sources with all their columns the same and the detectors with all their rows the same.

IV. Once the $M \times N$ differences have been calculated, it is trivial to generate the $M \times N$ values of **rj** (see Equation (1.8)). Then the program sums over the **M** different sources to get the final values of **E(0)** and **E(τ/4)** at each of the detectors (note the MATLAB function **sum, you can get the information by using the command of “help sum”**). Squaring and adding these gives you the intensities at the **N** detectors. Change the numbers **M** and **N** to something more reasonable, say about 51 sources and 201 detectors, and save & run the function again. Plot **Itot** versus **yd** and draw the final intensity distribution.

V. You need to refer back to this graph later, in the further labs. So save a copy of it, as you did in CO_1.3. Name the new saved file **CO_lab1_fig2.fig**.