A JAVA program for designing synchronously tuned extended cavity diode laser in Littrow configuration

Sarika Pugla & Santa Chawla
National Physical Laboratory, Dr K S Krishnan Road, New Delhi 110 012

Received 21 November 2002; accepted 22 August 2003

Extended cavity diode lasers (ECDL) in Littrow configuration have been widely used in precision atomic physics experiments world over. Designing the cavity to obtain a mode hop free operation in the desired tuning range requires synchronous tuning of the cavity and is extremely critical. A program in JAVA has been developed, taking into account the critical theoretical requirements of the ECDL. The software is user-friendly and the output provides the cavity design. The software can be used for any wavelength and grating with any number of lines in the Littrow configuration.

[Keywords: Extended cavity diode laser; Diode laser, Littrow configuration]

1 Introduction

Free running diode lasers have large linewidth. Though frequency tuning is possible by controlling temperature and injection current of diode laser, they often suffer from mode hop—a phenomenon which means sudden jump of the output frequency to a nearby oscillating mode of the diode cavity. The above facts make the use of free running diode lasers difficult for precision physics experiments and applications. The usual requirements from the laser are narrow linewidth, tunability to a particular reference frequency—mostly an atomic or molecular transition frequency and mode hop free scanning over a certain frequency range. Narrow linewidth can be obtained by optical feedback in an extended cavity, where the feedback element is a frequency selective component, e.g. a grating and such extended cavity diode lasers (ECDL) are extensively used.

In ECDL in Littrow configuration, a grating is placed in the Littrow angle as the feedback element and extended cavity is formed between the rear facet of the diode laser and the grating. However, synchronous tuning of the cavity mode and the feedback wavelength is essential for mode hop free scanning and is very critical for designing and fabrication of ECDL. The Java program that has been developed for this purpose takes into consideration such critical physical aspects for designing a Littrow cavity and can be used to design ECDL with varied parameters, e.g. diode laser wavelength, number of lines of grating, cavity length, collimating lens focal length etc. Java compiler generates byte code instructions that can be implemented on any machine.

Fig. 1 — Sketch of Littrow cavity showing diode laser (D), collimating lens, Grating (G). L is the cavity length before the rotation of grating starts. When the grating is rotated about an axis passing through R₂, the changed position of the grating is also shown.
Enter the Following Parameters

- Wavelength (nm): 852
- No. of lines (mm): 729
- Length of cavity (cm): 0
- Focal length of the lens (mm): 8
- Dimension of lens (mm): 0

\[ \theta = \frac{\Theta}{n} \times \sin^{-1} \left( \frac{n}{2} \right) \]  

\[ \lambda_c = 2L/N \]

Before rotation of the grating, selected wavelength of the grating is:

\[ \lambda_G = 2d \sin \theta \]

\( d \) being the grating pitch and \( \theta \) the angle of incidence at the grating, \( \lambda_G \) is the same as the cavity mode \( \lambda_c \), where:

\[ \lambda_c = 2L/N \]

\( L \) being the cavity length and \( N \) an integer. As the grating is rotated, the angle changes and \( \lambda_G \), changes. Since the cavity length changes, \( \lambda_c \) also changes. In order to have mode hop free tuning, it is necessary that \( \lambda_G \) and \( \lambda_c \) change synchronously.

2 Synchronous Tuning of Littrow Cavity

For designing a Littrow cavity, it is essential that optical feedback condition is fulfilled and single mode tuning over the scan range is obtained. For scanning, a PZT is attached to the grating. In a laser cavity, the oscillating mode depends upon the geometrical length of the cavity and the feedback wavelength depends upon the angle that grating makes with the laser beam. Therefore, tuning the laser cavity to a specific reference frequency depends critically on the grating angle. Continuous frequency scan is possible when a synchronous change in cavity length and grating angle is achieved by applying a voltage to PZT. Before rotation of the grating, selected wavelength of the grating is:

\[ \lambda_G = 2d \sin \theta \]

\[ \lambda_c = 2L/N \]

In a Littrow cavity, the optical length of the cavity is taken as the length measured from the emission point along the symmetry axis of the laser beam ending at the grating. The complete round trip phase accrued must be constantly independent of grating angle \( \theta \) and hence the feedback wavelength, during scanning so that, phase of feedback wave
remains the same with respect to the emitted wave. This condition ensures a synchronous feedback and cavity mode tuning during scanning by PZT attached to the grating. The calculations\(^1\) showed that, the best rotation axis of grating passes through the intersection point joining the grating plane and the laser, as shown in Fig. 1, and is parallel to the rulings of the grating. With this arrangement, it is possible to tune the laser output frequency over several nm without any mode hop. The calculation\(^2\) showed that, if the grating rotation point was fixed anywhere on the line \(R_1R_2\), mode hop suppression in the first order can be obtained. Any point lying on the line \(R_1R_2\) would provide an optimum rotation point, the best point being the intersection point \(R_2\).

3 Program

```java
import java.io.*;
import java.applet.*;
import java.awt.*;
import java.awt.event.*;
import java.util.*;
import java.lang.*;

/*<applet code="MountN.class" width="600" height="600">
</applet>*/

public class MountN extends Applet implements ActionListener {
    TextField t1,t2,t3,t4,t5;
    double Deg,B,H,y,lambda,N,Theta;
    int P1=0,B1=0,H1=0,h1=0,b1=0,p1=0;
    String s1,s2,s3,s4,s5;
    Double Lambda= new Double(0);
    Double n= new Double(0);
    Double P= new Double(0);
    Double P= new Double(0);
    Double f= new Double(0);
    Double dim=new Double(0);
    Button but1;
    public void init()
    {
        setLayout(null);
        t1=new TextField(30);
        t2=new TextField(30);
        t3=new TextField(30);
        t4=new TextField(30);
        t5=new TextField(30);
        but1=new Button("Enter");
        but1.setBounds(160,180,50,20);
        add(but1);
        but1.addActionListener(this);
        t1.setBounds(210,55,50,20);
        add(t1);
        t2.setBounds(210,80,50,20);
        add(t2);
        t3.setBounds(210,105,50,20);
        add(t3);
        t4.setBounds(210,130,50,20);
        add(t4);
        t5.setBounds(210,155,50,20);
        add(t5);
    }
```
/* Entering the value of different parameters, viz., wavelength, number of lines per mm in the grating, length of the cavity, back focal length of the lens, dimension of lens */

public void paint(Graphics g)
{
    g.drawString("Enter the Following Parameters: ",25,50);
    g.drawString("Wavelength(mm)",25,70);
    g.drawString("No of lines(/mm) ",25,95);
    g.drawString("Length of the cavity(cm),25,120);
    g.drawString("Focal length of the lens(mm),25,145);
    g.drawString("Dimension of lens(mm)",25,170);
    try
    {
        s1=t1.getText();
        Lambda=Double.valueOf(s1);
        s2=t2.getText();
        n=Double.valueOf(s2);
        s3=t3.getText();
        P=Double.valueOf(s3);
        s4=t4.getText();
        length=Double.valueOf(s4);
        s5=t5.getText();
        dim=Double.valueOf(s5);
    } catch(Exception e){}
    lambda=Lambda.doubleValue();
    N=n.doubleValue();

/* Calculation of angle theta using formulas stated above */
y=(lambda*N*(1e-6))/2;
Theta=(180*Math.asin(y)/Math.PI);
B=P.doubleValue()/Math.tan(Math.asin(y));
H=P.doubleValue()/Math.sin(Math.asin(y));

/* Drawing the sketch of the cavity (1mm= 2 Pixels)*/
P1=(int)Math.round(P.doubleValue()*20);
B1=(int)Math.round(B*20);
H1=(int)Math.round(H*20);
if(B1>0)
{
    g.drawString("Angle Theta = ",270,70);
    s4=Double.toString(Theta);
    g.drawString(s4,350,70);
    g.setColor(Color.pink);
    g.fillRect(100,303),36);
    g.setColor(Color.blue);
    g.drawString("D",90,321);
    g.setColor(Color.red);
    g.drawLine(108,321,108+P1,321);
    g.setColor(Color.black);
    g.drawString("G",110+P1,321);
    g.drawLine(108,321,108+P1,321);
    g.setColor(Color.red);
    g.drawString("R2",108,(319-B1));
    g.setColor(Color.black);
    g.drawString("R1",108,(321-B1));
    g.setColor(Color.red);
    g.drawLine(108+P1,321,108,321+B1));
public void actionPerformed(ActionEvent e)
{
    repaint();
}
/*End of the Program*/

4 Executing the Program

For the execution of the program three files will be needed, viz.,

- Java File: The program will be stored in this file. (save as *.java)
- Class file: This file will be generated using Javac, Java compiler. Command is:
  javac *.java.
- HTML file: This file will include the HTML code for the applet.

All these files are saved in the same folder.

To run the program using Internet Explorer*, html is opened which needs only the class file for the execution. While, for the execution of the program using appletviewer Java file and class file will be needed.

The output window and flow chart are presented in Figs 2 and 3, respectively.

5 Conclusions

An extended cavity diode laser (ECDL), in Littrow configuration has been designed and fabricated. Synchronous tuning of cavity mode with PZT displacement has been observed. The ECDL frequency was tuned to the transition frequency of Cs atom at 852 nm and active stabilization of ECDL frequency against Cs transition frequency was achieved.

References