Article

3D visualization of objects and molecules: An integrative Java software

WenJun Zhang
School of Life Sciences, Sun Yat-sen University, Guangzhou, China
E-mail: zhwj@mail.sysu.edu.cn, wjzhang@iaees.org

Received 20 March 2023; *Accepted 16 April 2023*; *Published online 15 May 2023*; *Published 1 December 2023*

Abstract
An integrative Java software, visual3D 1.0, for 3D visualization of objects and chemical molecules was developed in present study. The software uses a XYZ or OBJ file to generate the 3D graphics represented by the file. Various parameters can be specified by users. In the generated 3D graphics window, users may right- or left-click mouse to zoom in or zoom out the 3D graphics, or mouse-drag the graphics to rotate the 3D graphics, or slide scrollbar to vertically or horizontally translate the 3D graphics. The 3D graphics can be saved as an image file (in PNG, JPG, or other formats) at its current appearance. Both visual3D 1.0 and demonstration data files were given.

Keywords 3D visualization; objects; chemical molecules; Java; software; integration.

Computational Ecology and Software
ISSN 2220-721X
URL: http://www.iaees.org/publications/journals/ces/online-version.asp
RSS: http://www.iaees.org/publications/journals/ces/rss.xml
E-mail: ces@iaees.org
Editor-in-Chief: WenJun Zhang
Publisher: International Academy of Ecology and Environmental Sciences

1 Introduction

Digital visualization is important for scientific research and applications (Narad et al., 2017). So far numerous visualization software have been developed, e.g., the software and tools for network visualization (Zhang, 2007, 2012a-b, 2016, 2018, 2021, 2024a-d). Some of them were developed as Java tools (Zhang, 2011a-c; Zhang, 2012a-b; Zhang and Zheng, 2012; Zhang et al., 2015). Java is powerful for developing platform independent tools and thus be widely used. In present study, an integrative Java software, visual3D 1.0, for 3D visualization of objects and chemical molecules was developed based on JDK 1.1 and J2SDK 1.4.2. The software can use an XYZ or OBJ file to generate the 3D graphics represented by the file. Some Java codes, the executable software and demonstration data files are given.

2 Software and Data

2.1 Software and runtime environment

The Java software, visual3D version 1.0, was further developed based on JDK 1.1 (Java Development Kit) and J2SDK1.4.2. Seven Java classes, graphicsApplet, graphicsPanel, Model3D, Mat3D, Atom, paraInput, graphicsFrame (Zhang, 2007, 2012a, 2024b) are included and loaded by the class, visual3D. The following are

Java codes of the class visual3D:

```
import java.awt.*;
import java.io.*;

public class visual3D extends Frame
{

    public visual3D()
    {
        byte byte0 = 15;
        int i = 1;
        panel = new Panel();
        gbl = new GridBagLayout();
        gbc = new GridBagConstraints();
        panel.setLayout(gbl);
        gbc.fill = 1;
        cbg = new CheckboxGroup();
        cb1 = new Checkbox(" ", cbg, false);
        cb2 = new Checkbox(" ", cbg, true);
        label = new Label("Object Type: ");
        cb1.setLabel("Molecule");
        cb2.setLabel("Wire Object");
        layout(0, 0, byte0, i, label);
        layout(0, 1, byte0, i, cb2);
        layout(byte0, 1, byte0 + byte0, i, cb1);
        label1 = new Label("Scale (3, 5, etc.): ");
        edit1 = new TextField("2");
        label2 = new Label("Dilation Rate of Scale (0.5, 1, etc.): ");
        edit2 = new TextField("0.3");
        label4 = new Label("Maximum Scroll (1000, 10000, etc.): ");
        edit6 = new TextField("2000");
        layout(0, 2, byte0, i, label1);
        layout(byte0, 2, byte0 + byte0, i, edit1);
        layout(0, 3, byte0, i, label2);
        layout(byte0, 3, byte0 + byte0, i, edit2);
        layout(0, 4, byte0, i, label4);
        layout(byte0, 4, byte0 + byte0, i, edit6);
        label7 = new Label("Canvas Width: ");
        label8 = new Label("Canvas Height: ");
        edit4 = new TextField("900");
        edit5 = new TextField("680");
        layout(0, 5, byte0, i, label7);
        layout(byte0, 5, byte0 + byte0, i, edit4);
        layout(0, 6, byte0, i, label8);
        layout(byte0, 6, byte0 + byte0, i, edit5);
        buttonok = new Button("OK");
        buttonclose = new Button("Close");
        layout(0, 7, byte0 + byte0, 2 * i, buttonok);
```

```
        layout(byte0, 7, byte0 + byte0, i, buttonclose);
        add(panel);
        resize(300, 650);
        setLocation(200, 80);
        pack();
        show();
    }

    public void layout(int i, int j, int k, int l, Component component)
    {
        gbc.gridx = i;
        gbc.gridy = j;
        gbc.gridwidth = k;
        gbc.gridheight = l;
        gbl.setConstraints(component, gbc);
        panel.add(component);
    }

    public boolean action(Event event, Object obj)
    {
        if(event.target == cb1)
        {
            edit1.setText("10");
            vector = new String[3];
            vector[1] = "Size of Balls";
            vector[2] = "Number of Ball Colors";
            parainput = new paraInput(1, 2, vector);
            parainput.parameterinput[1].setText("10");
            parainput.parameterinput[2].setText("20");
            buttonok.setEnabled(false);
            return true;
        }
        if(event.target == buttonclose)
            System.exit(0);
        if(event.target == buttonok)
        {
            if(cbg.getCurrent() == cb1)
                sele = 1;
            else
            if(cbg.getCurrent() == cb2)
                sele = 0;
            scale = Float.valueOf(edit1.getText().trim()).floatValue();
            scaledila = Float.valueOf(edit2.getText().trim()).floatValue();
            if(sele == 1)
            {
                ballsize = Float.valueOf(parainput.vec[1]).floatValue();
                ballcol = Integer.valueOf(parainput.vec[2]).intValue();
            }
            scrollmax = Integer.valueOf(edit6.getText().trim()).intValue();
            canvaswid = Integer.valueOf(edit4.getText().trim()).intValue();
```

```
                canvashei = Integer.valueOf(edit5.getText().trim()).intValue();
                Frame frame = new Frame();
                String s = "";
                if(sele == 1)
                    s = "Open data file (*.xyz)";
                else
                if(sele == 0)
                    s = "Open data file (*.obj)";
                FileDialog filedialog = new FileDialog(frame, s, 0);
                if(sele == 1)
                    filedialog.setFile("*.xyz");
                else
                if(sele == 0)
                    filedialog.setFile("*.obj");
                frame.setLocation(250, 150);
                filedialog.show();
                String s1 = filedialog.getDirectory() + filedialog.getFile();
                try
                {
                    inputstream = new FileInputStream(s1);
                }
                catch(Exception exception) { }
                if(sele == 1)
                    (new graphicsFrame(new graphicsPanel(inputstream, sele, scale, scaledila, scrollmax, ballsize, ballcol,
canvaswid, canvashei), "Zhang WJ. 2023. 3D visualization of objects and molecules: An integrative Java software.
Computational Ecology and Software, 13(4): 81-108")).resize(canvaswid, canvashei);
                else
                if(sele == 0)
                    (new graphicsFrame(new graphicsPanel(inputstream, sele, scale, scaledila, scrollmax, canvaswid,
canvashei), "Zhang WJ. 2023. 3D visualization of objects and molecules: An integrative Java software. Computational Ecology
and Software, 13(4): 81-108")).resize(canvaswid, canvashei);
                return true;
            } else
            {
                return false;
            }
        }

        public static void main(String args[])
            throws IOException
        {
            new visual3D();
        }

        InputStream inputstream;
        paraInput parainput;
        String vector[];
        int ballcol;
        int sele;
        int scrollmax;
```

```
    float ballsize;
    float scale;
    float scaledila;
    int canvaswid;
    int canvashei;
    CheckboxGroup cbg;
    Checkbox cb1;
    Checkbox cb2;
    static Button buttonok;
    Button buttonclose;
    TextField edit1;
    TextField edit2;
    TextField edit6;
    TextField edit4;
    TextField edit5;
    Label label;
    Label label1;
    Label label2;
    Label label4;
    Label label7;
    Label label8;
    Panel panel;
    GridBagLayout gbl;
    GridBagConstraints gbc;
}
```

The following are Java codes of the class graphicsPanel:

```
import java.applet.Applet;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.image.BufferedImage;
import java.io.File;
import java.io.InputStream;
import javax.imageio.ImageIO;

public class graphicsPanel extends Applet
    implements ActionListener
{

    public graphicsPanel(InputStream inputstream, int i, float f, float f1, int j, float f2, int k,
            int l, int i1)
    {
        is = null;
        is = inputstream;
        sele = i;
        scale = f;
        scaledila = f1;
```

```
        scrollmax = j;
        ballsize = f2;
        ballcol = k;
        canvaswid = l;
        canvashei = i1;
        setGraphicsApplet();
    }

    public graphicsPanel(InputStream inputstream, int i, float f, float f1, int j, int k, int l)
    {
        is = null;
        is = inputstream;
        sele = i;
        scale = f;
        scaledila = f1;
        scrollmax = j;
        canvaswid = k;
        canvashei = l;
        setGraphicsApplet();
    }

    public void setGraphicsApplet()
    {
        if(sele == 1)
            gapplet = new graphicsApplet(is, sele, scale, scaledila, scrollmax, ballsize, ballcol);
        else
        if(sele == 0)
            gapplet = new graphicsApplet(is, sele, scale, scaledila, scrollmax);
        save = new Button("Save");
        close = new Button("Close");
        setLayout(new BorderLayout());
        add("Center", gapplet);
        controlPanel = new Panel();
        add("South", controlPanel);
        controlPanel.add(save);
        controlPanel.add(close);
        save.addActionListener(this);
        close.addActionListener(this);
        resize(canvaswid - 15, canvashei - 15);
        setLocation(20, 20);
        validate();
        show();
    }

    public void init()
    {
    }

    public void paint(Graphics g)
    {
```

```
        repaint();
    }

    public void stop()
    {
    }

    public void destroy()
    {
        remove(gapplet);
    }

    public void actionPerformed(ActionEvent actionevent)
    {
        Object obj = actionevent.getSource();
        if(obj == save)
        {
            try
            {
                Frame frame = new Frame();
                FileDialog filedialog = new FileDialog(frame, "Save image as (*.png, *.jpg, etc)", 1);
                filedialog.setFile("*.png");
                filedialog.show();
                String s = filedialog.getDirectory() + filedialog.getFile();
                buffered = toBufferedImage(gapplet.offscreen);
                ImageIO.write(buffered, "png", new File(s));
            }
            catch(Exception exception) { }
            gapplet.scrbarver.removeAdjustmentListener(gapplet.l1);
            gapplet.scrbarhor.removeAdjustmentListener(gapplet.l2);
            remove(gapplet);
            System.gc();
            getParent().hide();
        }
        if(obj == close)
        {
            gapplet.scrbarver.removeAdjustmentListener(gapplet.l1);
            gapplet.scrbarhor.removeAdjustmentListener(gapplet.l2);
            remove(gapplet);
            System.gc();
            getParent().hide();
        }
    }

    public BufferedImage toBufferedImage(Image image)
    {
        if(image instanceof BufferedImage)
        {
            return (BufferedImage)image;
        } else
```

```
            {
                BufferedImage bufferedimage = new BufferedImage(image.getWidth(null), image.getHeight(null), 1);
                Graphics2D graphics2d = bufferedimage.createGraphics();
                graphics2d.drawImage(image, 0, 0, null);
                graphics2d.dispose();
                return bufferedimage;
            }
        }

        float scale;
        float scaledila;
        float ballsize;
        int sele;
        int scrollmax;
        int ballcol;
        int canvaswid;
        int canvashei;
        InputStream is;
        Button close;
        Button save;
        Panel controlPanel;
        graphicsApplet gapplet;
        BufferedImage buffered;
}
```

The following are Java codes of the class graphicsApplet:

```
import java.applet.Applet;
import java.awt.*;
import java.awt.event.*;
import java.io.InputStream;

public class graphicsApplet extends Applet
        implements Runnable, MouseListener, MouseMotionListener
{

        public graphicsApplet()
        {
                painted = true;
                is = null;
                message = null;
                md = null;
                model = null;
                offscreen = null;
                amat = new Mat3D();
                tmat = new Mat3D();
                amat.xrot(20D);
                amat.yrot(20D);
                resize(getSize().width, getSize().height);
```

```
        addMouseListener(this);
        addMouseMotionListener(this);
    }

    public graphicsApplet(InputStream inputstream, int i, float f, float f1, int j, float f2, int k)
    {
        this();
        is = inputstream;
        sele = i;
        scale = f;
        scaledila = f1;
        scrollmax = j;
        ballsize = f2;
        ballcol = k;
        scrollverpos = scrollhorpos = scrollmidpos = (int)((double)scrollmax / 2D + 0.5D);
        scrbarhor = new Scrollbar(0, scrollhorpos, 20, 0, scrollmax);
        scrbarver = new Scrollbar(1, scrollverpos, 20, 0, scrollmax);
        setLayout(new BorderLayout());
        add("South", scrbarhor);
        add("East", scrbarver);
        (new Thread(this)).start();
    }

    public graphicsApplet(InputStream inputstream, int i, float f, float f1, int j)
    {
        this();
        is = inputstream;
        sele = i;
        scale = f;
        scaledila = f1;
        scrollmax = j;
        scrollverpos = scrollhorpos = scrollmidpos = (int)((double)scrollmax / 2D + 0.5D);
        scrbarhor = new Scrollbar(0, scrollhorpos, 20, 0, scrollmax);
        scrbarver = new Scrollbar(1, scrollverpos, 20, 0, scrollmax);
        setLayout(new BorderLayout());
        add("South", scrbarhor);
        add("East", scrbarver);
        (new Thread(this)).start();
    }

    public void newoffscreen()
    {
        offscreen = createImage(getSize().width, getSize().height);
        offgraphics = offscreen.getGraphics();
    }

    public void run()
    {
        l1 = new AdjustmentListener() {
```

```
        public void adjustmentValueChanged(AdjustmentEvent adjustmentevent)
        {
            posver = graphicsApplet.scrbarver.getValue();
            dify = posver - scrollverpos;
            difabsy = posver - scrollmidpos;
            scrollverpos = posver;
            amat.translate(0.0F, dify, 0.0F);
            if(painted)
            {
                painted = false;
                repaint();
            }
        }

    };
    scrbarver.addAdjustmentListener(l1);
    l2 = new AdjustmentListener() {

        public void adjustmentValueChanged(AdjustmentEvent adjustmentevent)
        {
            poshor = graphicsApplet.scrbarhor.getValue();
            difx = poshor - scrollhorpos;
            difabsx = poshor - scrollmidpos;
            scrollhorpos = poshor;
            amat.translate(-difx, 0.0F, 0.0F);
            if(painted)
            {
                painted = false;
                repaint();
            }
        }

    };
    scrbarhor.addAdjustmentListener(l2);
    try
    {
        Thread.currentThread().setPriority(1);
        if(sele == 1)
        {
            model = new Model3D(is, sele, ballcol, ballsize);
            Atom.setApplet(this);
        } else
        if(sele == 0)
            model = new Model3D(is, sele);
        md = model;
        model.findBB();
        if(sele == 0)
            model.compress();
        float f = model.xmax - model.xmin;
        float f1 = model.ymax - model.ymin;
```

```
            float f2 = model.zmax - model.zmin;
            if(f1 > f)
                f = f1;
            if(f2 > f)
                f = f2;
            float f3 = (float)getSize().width / f;
            float f4 = (float)getSize().height / f;
            xfac = 0.7F * (f3 >= f4 ? f4 : f3) * scale;
            xfac0 = xfac;
        }
        catch(Exception exception)
        {
            if(sele == 1)
                exception.printStackTrace();
            md = null;
            message = exception.toString();
        }
        try
        {
            if(is != null)
                is.close();
        }
        catch(Exception exception1) { }
        repaint();
    }

    public void start()
    {
    }

    public void stop()
    {
    }

    public void destroy()
    {
        scrbarver.removeAdjustmentListener(l1);
        scrbarhor.removeAdjustmentListener(l2);
        removeMouseListener(this);
        removeMouseMotionListener(this);
        System.gc();
    }

    public void mouseClicked(MouseEvent mouseevent)
    {
        if(mouseevent.getButton() == 1)
            xfac = xfac + xfac * scaledila;
        else
        if((mouseevent.getButton() == 2) | (mouseevent.getButton() == 3))
            if(1.0F - scaledila > 0.0F)
```

```
                xfac = xfac - xfac * scaledila;
            else
                xfac = 0.1F;
        if(painted)
        {
            painted = false;
            repaint();
        }
    }

    public void mousePressed(MouseEvent mouseevent)
    {
        xfa = xfac / xfac0;
        prevx = (int)((float)mouseevent.getX() * xfa);
        prevy = (int)((float)mouseevent.getY() * xfa);
        mouseevent.consume();
    }

    public void mouseReleased(MouseEvent mouseevent)
    {
    }

    public void mouseEntered(MouseEvent mouseevent)
    {
    }

    public void mouseExited(MouseEvent mouseevent)
    {
    }

    public void mouseDragged(MouseEvent mouseevent)
    {
        int i = (int)((float)mouseevent.getX() * xfa);
        int j = (int)((float)mouseevent.getY() * xfa);
        tmat.unit();
        float f = (float)(prevy - j) * (360F / ((float)getSize().width * xfa));
        float f1 = (float)(i - prevx) * (360F / ((float)getSize().height * xfa));
        tmat.xrot(f);
        tmat.yrot(f1);
        amat.mult(tmat);
        if(painted)
        {
            painted = false;
            repaint();
        }
        prevx = i;
        prevy = j;
        mouseevent.consume();
    }
```

```
public void mouseMoved(MouseEvent mouseevent)
{
}

public synchronized void update(Graphics g)
{
    if(offscreen == null)
        g.clearRect(0, 0, getSize().width, getSize().height);
    paint(g);
}

public void paint(Graphics g)
{
    if(offscreen == null)
        newoffscreen();
    if(md != null)
    {
        md.mat.unit();
        md.mat.translate(-(md.xmin + md.xmax) / 2.0F, -(md.ymin + md.ymax) / 2.0F, -(md.zmin + md.zmax) / 2.0F);
        md.mat.mult(amat);
        md.mat.scale(xfac, -xfac, (16F * xfac) / (float)getSize().width);
        md.mat.translate(getSize().width / 2, getSize().height / 2, 8F);
        md.transformed = false;
        offgraphics.setColor(getBackground());
        offgraphics.fillRect(0, 0, getSize().width, getSize().height);
        md.paint(offgraphics);
        g.drawImage(offscreen, 0, 0, this);
        setPainted();
    } else
    if(message != null)
    {
        g.drawString("Error in model:", 3, 20);
        g.drawString(message, 10, 40);
    }
}

private synchronized void setPainted()
{
    painted = true;
    notifyAll();
}

Model3D md;
Model3D model;
boolean painted;
float xfac;
float xfac0;
float xfa;
int prevx;
int prevy;
```

```
    float xtheta;
    float ytheta;
    Mat3D amat;
    Mat3D tmat;
    String message;
    Image offscreen;
    Graphics offgraphics;
    float scale;
    float scaledila;
    float ballsize;
    int sele;
    int scrollmax;
    int scrollmidpos;
    int scrollverpos;
    int scrollhorpos;
    int posver;
    int poshor;
    int difx;
    int dify;
    int difabsx;
    int difabsy;
    int ballcol;
    InputStream is;
    static Scrollbar scrbarver;
    static Scrollbar scrbarhor;
    static AdjustmentListener l1;
    static AdjustmentListener l2;
}
```

The following are Java codes of the class Model3D:

```
import java.awt.Color;
import java.awt.Graphics;
import java.io.*;
import java.util.Hashtable;

class Model3D
{

    Model3D()
        throws Exception
    {
        mat = new Mat3D();
        mat.xrot(20D);
        mat.yrot(30D);
    }

    Model3D(InputStream inputstream, int i)
        throws Exception
```

```
    {
        StreamTokenizer streamtokenizer;
label0:
        {
            this();
            is = inputstream;
            sele = i;
            streamtokenizer = new StreamTokenizer(new BufferedReader(new InputStreamReader(is)));
            streamtokenizer.eolIsSignificant(true);
            streamtokenizer.commentChar(35);
            try
            {
label1:
                do
                {
label2:
                    {
label3:
                        do
                        {
label4:
                            do
                                switch(streamtokenizer.nextToken())
                                {
                                default:
                                    break label3;

                                case 10: // '\n'
                                    break;

                                case -3:
                                    if(!"v".equals(streamtokenizer.sval))
                                        break label4;
                                    double d = 0.0D;
                                    double d1 = 0.0D;
                                    double d2 = 0.0D;
                                    if(streamtokenizer.nextToken() == -2)
                                    {
                                        d = streamtokenizer.nval;
                                        if(streamtokenizer.nextToken() == -2)
                                        {
                                            d1 = streamtokenizer.nval;
                                            if(streamtokenizer.nextToken() == -2)
                                                d2 = streamtokenizer.nval;
                                        }
                                    }
                                    addVert((float)d, (float)d1, (float)d2);
                                    while(streamtokenizer.ttype != 10 && streamtokenizer.ttype != -1)
                                        streamtokenizer.nextToken();
                                    break;
```

```
                                    }
                                while(true);
                                if(!"f".equals(streamtokenizer.sval) && !"fo".equals(streamtokenizer.sval)
&& !"l".equals(streamtokenizer.sval))
                                    break label2;
                                int j = -1;
                                int k = -1;
                                byte byte0 = -1;
                                do
                                {
                                    while(streamtokenizer.nextToken() == -2)
                                    {
                                        int l = (int)streamtokenizer.nval;
                                        if(k >= 0)
                                            add(k - 1, l - 1);
                                        if(j < 0)
                                            j = l;
                                        k = l;
                                    }
                                    if(streamtokenizer.ttype != 47)
                                        break;
                                    streamtokenizer.nextToken();
                                } while(true);
                                if(j >= 0)
                                    add(j - 1, k - 1);
                            } while(streamtokenizer.ttype == 10);
                            break label1;
                        }
                        while(streamtokenizer.nextToken() != 10 && streamtokenizer.ttype != -1) ;
                    } while(true);
                    is.close();
                    break label0;
                }
                catch(Exception exception) { }
            }
            if(streamtokenizer.ttype != -1)
                throw new Exception(streamtokenizer.toString());
            else
                return;
        }

        Model3D(InputStream inputstream, int i, int j, float f)
            throws Exception
        {
            this();
            is = inputstream;
            sele = i;
            ballcol = j;
            ballsize = f;
            StreamTokenizer streamtokenizer = new StreamTokenizer(new BufferedReader(new InputStreamReader(is)));
```

```
        streamtokenizer.eolIsSignificant(true);
        streamtokenizer.commentChar(35);
        try
        {
label0:
            do
                switch(streamtokenizer.nextToken())
                {
                default:
                    break;

                case -3:
                    String s = streamtokenizer.sval;
                    double d = 0.0D;
                    double d1 = 0.0D;
                    double d2 = 0.0D;
                    if(streamtokenizer.nextToken() == -2)
                    {
                        d = streamtokenizer.nval;
                        if(streamtokenizer.nextToken() == -2)
                        {
                            d1 = streamtokenizer.nval;
                            if(streamtokenizer.nextToken() == -2)
                                d2 = streamtokenizer.nval;
                        }
                    }
                    addVert(s, (float)d, (float)d1, (float)d2);
                    while(streamtokenizer.ttype != 10 && streamtokenizer.ttype != -1)
                        streamtokenizer.nextToken();
                    break;

                case -1:
                    is.close();
                    break label0;
                }
            while(true);
        }
        catch(Exception exception) { }
        if(streamtokenizer.ttype != -1)
            throw new Exception(streamtokenizer.toString());
        else
            return;
    }

    int addVert(String s, float f, float f1, float f2)
    {
        int i = nvert;
        if(i >= maxvert)
            if(vert == null)
            {
```

```
                maxvert = 0xf4240;
                vert = new float[maxvert * 3];
                atoms = new Atom[maxvert];
            } else
            {
                maxvert *= 2;
                float af[] = new float[maxvert * 3];
                System.arraycopy(vert, 0, af, 0, vert.length);
                vert = af;
            }
        Atom atom = (Atom)atomTable.get(s.toLowerCase());
        if(atom == null)
            atom = defaultAtom;
        atoms[i] = atom;
        i *= 3;
        vert[i] = f;
        vert[i + 1] = f1;
        vert[i + 2] = f2;
        return nvert++;
    }

    int addVert(float f, float f1, float f2)
    {
        int i = nvert;
        if(i >= maxvert && vert == null)
        {
            maxvert = 0xf4240;
            vert = new float[maxvert * 3];
        }
        i *= 3;
        vert[i] = f;
        vert[i + 1] = f1;
        vert[i + 2] = f2;
        return nvert++;
    }

    void add(int i, int j)
    {
        int k = ncon;
        if(i >= nvert || j >= nvert)
            return;
        if(k >= maxcon && con == null)
        {
            maxcon = 0xf4240;
            con = new int[maxcon];
        }
        if(i > j)
        {
            int l = i;
            i = j;
```

```
            j = l;
        }
        con[k] = i << 16 | j;
        ncon = k + 1;
    }

    void transform()
    {
        if(transformed || nvert <= 0)
            return;
        if(tvert == null || tvert.length < nvert * 3)
            tvert = new int[nvert * 3];
        mat.transform(vert, tvert, nvert);
        transformed = true;
    }

    void quickSort(int ai[], int i, int j)
    {
        int k = i;
        int l = j;
        if(j > i)
        {
            int i1 = ai[(i + j) / 2];
            do
            {
                if(k > l)
                    break;
                for(; k < j && ai[k] < i1; k++);
                for(; l > i && ai[l] > i1; l--);
                if(k <= l)
                {
                    swap(ai, k, l);
                    k++;
                    l--;
                }
            } while(true);
            if(i < l)
                quickSort(ai, i, l);
            if(k < j)
                quickSort(ai, k, j);
        }
    }

    void swap(int ai[], int i, int j)
    {
        int k = ai[i];
        ai[i] = ai[j];
        ai[j] = k;
    }
```

```
void compress()
{
    int i = ncon;
    int ai[] = con;
    quickSort(con, 0, ncon - 1);
    int j = 0;
    int k = -1;
    for(int l = 0; l < i; l++)
    {
        int i1 = ai[l];
        if(k != i1)
        {
            ai[j] = i1;
            j++;
        }
        k = i1;
    }

    ncon = j;
}

void paint(Graphics g)
{
    if(sele == 1)
    {
        if(vert == null || nvert <= 0)
            return;
        transform();
        int ai[] = tvert;
        int ai1[] = ZsortMap;
        if(ai1 == null)
        {
            ai1 = new int[nvert];
            for(int i1 = nvert; --i1 >= 0;)
                ai1[i1] = i1 * 3;
        }
        int j1 = nvert - 1;
        boolean flag;
        do
        {
            if(--j1 < 0)
                break;
            flag = false;
            for(int l1 = 0; l1 <= j1; l1++)
            {
                int l2 = ai1[l1];
                int k3 = ai1[l1 + 1];
                if(ai[l2 + 2] > ai[k3 + 2])
                {
                    ai1[l1 + 1] = l2;
```

```
                    ai1[l1] = k3;
                    flag = true;
                }
            }

        } while(flag);
        j1 = 0;
        int i2 = nvert;
        if(i2 <= 0 || nvert <= 0)
            return;
        for(int i3 = 0; i3 < i2; i3++)
        {
            int l3 = ai1[i3];
            int j4 = ai[l3 + 2];
            if(j4 < 0)
                j4 = 0;
            if(j4 > 15)
                j4 = 15;
            atoms[l3 / 3].paint(g, ai[l3], ai[l3 + 1], j4, ballcol, ballsize);
        }

    }
    if(sele == 0)
    {
        if(vert == null || nvert <= 0)
            return;
        transform();
        if(gr == null)
        {
            gr = new Color[16];
            for(int i = 0; i < 16; i++)
            {
                int k = (int)(170D * (1.0D - Math.pow((double)i / 15D, 2.2999999999999998D)));
                gr[i] = new Color(k, k, k);
            }

        }
        int j = 0;
        int l = ncon;
        int ai2[] = new int[con.length];
        for(int k1 = 0; k1 < con.length; k1++)
            ai2[k1] = con[k1];

        int ai3[] = new int[tvert.length];
        for(int j2 = 0; j2 < tvert.length; j2++)
            ai3[j2] = tvert[j2];

        if(l <= 0 || nvert <= 0)
            return;
        for(int k2 = 0; k2 < l; k2++)
```

```
            {
                int j3 = ai2[k2];
                int i4 = (j3 >> 16 & 0xffff) * 3;
                int k4 = (j3 & 0xffff) * 3;
                int l4 = ai3[i4 + 2] + ai3[k4 + 2];
                if(l4 < 0)
                    l4 = 0;
                if(l4 > 15)
                    l4 = 15;
                if(l4 != j)
                {
                    j = l4;
                    g.setColor(gr[l4]);
                }
                g.drawLine(ai3[i4], ai3[i4 + 1], ai3[k4], ai3[k4 + 1]);
            }

        }
    }

    void findBB()
    {
        if(nvert <= 0)
            return;
        float af[] = new float[vert.length];
        for(int i = 0; i < vert.length; i++)
            af[i] = vert[i];

        float f = af[0];
        float f1 = f;
        float f2 = af[1];
        float f3 = f2;
        float f4 = af[2];
        float f5 = f4;
        int j = nvert * 3;
        do
        {
            if((j -= 3) <= 0)
                break;
            float f6 = af[j];
            if(f6 < f)
                f = f6;
            if(f6 > f1)
                f1 = f6;
            float f7 = af[j + 1];
            if(f7 < f2)
                f2 = f7;
            if(f7 > f3)
                f3 = f7;
            float f8 = af[j + 2];
```

```
                if(f8 < f4)
                    f4 = f8;
                if(f8 > f5)
                    f5 = f8;
            } while(true);
            xmax = f1;
            xmin = f;
            ymax = f3;
            ymin = f2;
            zmax = f5;
            zmin = f4;
        }

        float vert[];
        Atom atoms[];
        int tvert[];
        int con[];
        int ZsortMap[];
        int nvert;
        int ncon;
        int maxcon;
        int maxvert;
        int sele;
        int ballcol;
        float ballsize;
        InputStream is;
        static Hashtable atomTable;
        static Atom defaultAtom = new Atom(255, 100, 200);
        boolean transformed;
        Mat3D mat;
        float xmin;
        float xmax;
        float ymin;
        float ymax;
        float zmin;
        float zmax;
        static Color gr[];

        static
        {
            atomTable = new Hashtable();
            atomTable.put("c", new Atom(0, 0, 0));
            atomTable.put("h", new Atom(210, 210, 210));
            atomTable.put("n", new Atom(0, 0, 255));
            atomTable.put("o", new Atom(255, 0, 0));
            atomTable.put("p", new Atom(255, 0, 255));
            atomTable.put("s", new Atom(255, 255, 0));
            atomTable.put("hn", new Atom(150, 255, 150));
        }
    }
```

Pack all classes into a JAR package and compile the package into the executable Java software, visual3D 1.0.exe.

Before using the software, the Java Runtime Environment (JRE) should be installed. First, download the Java Runtime Environment software from the following website: https://www.java.com/en/. After downloading, click Install Java Runtime Environment to install it. Alternatively, search for "Java Runtime Environment" on the Internet to find the appropriate platform version for downloading. Then, double-click visual3D 1.0.exe to run the software (Fig. 1 and 2).

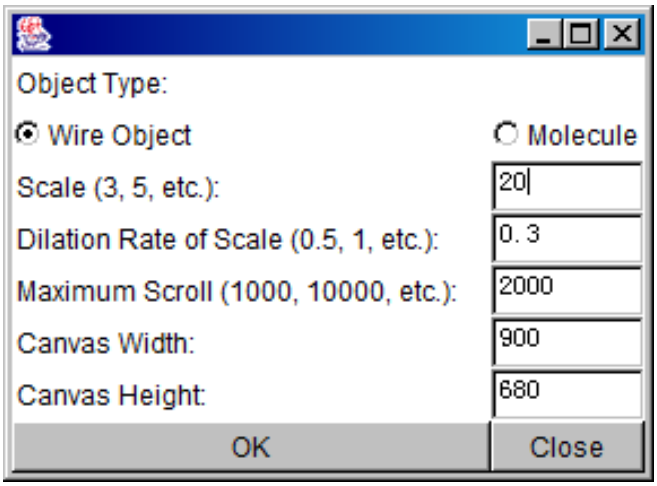

Fig. 1 The window interface of visual3D 1.0.

Open the software, make choices between object types and enter the parameters (default settings are used). If a molecule graphics is to be visualized (i.e., a XYZ file is to be used), additional parameters should be entered in a new window (Fig. 2). After chosing a data file (Fig. 3), the window for 3D graphics will be generated.

In the generated 3D graphics window (Figs 4-7), users may right- or left-click mouse to zoom in or zoom out the 3D graphics, or mouse-drag the graphics to rotate the 3D graphics, or slide scrollbar to vertically or horizontally translate the 3D graphics.

Finally, the 3D graphics can be saved as an image file (in PNG, JPG, or other formats) at its current appearance.

2.2 Data

The data files for 3D visualization of objects are OBJ and XYZ files. OBJ is a 3D model file format (*.obj). It is a type of text files, which can be opened with a text editor (e.g., notepad). Almost all 3D software supports OBJ files. XYZ is a file format (*.xyz) for recording structures of chemical molecules and other objects. It is a type of text files and can be opened with a text editor. Almost all chemical and molecular software supports XYZ files. Users may download free OBJ or XYZ files from internet resources for using the present software. There are numerous such files on the internet.

The software and demo data files (demo data were introduced from JDK 1.1) are included in the package: http://www.iaees.org/publications/journals/ces/articles/2023-13(4)/e-suppl/Zhang-Supplementary-Material.rar Users may occasionally examine and download available higher versions of the software in this package.

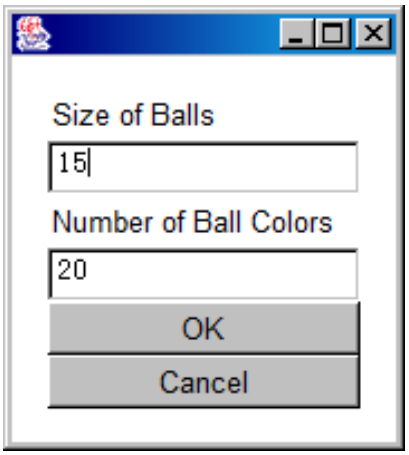

Fig. 2 The parameter input interface of visual3D 1.0 for XYZ files.

Fig. 3 The file open dialog of visual3D 1.0.

3 Example Demonstration

The XYZ and OBJ data for demonstration were from JDK 1.1. Some of the generated 3D graphics are indicated in Figs 4 to 7.

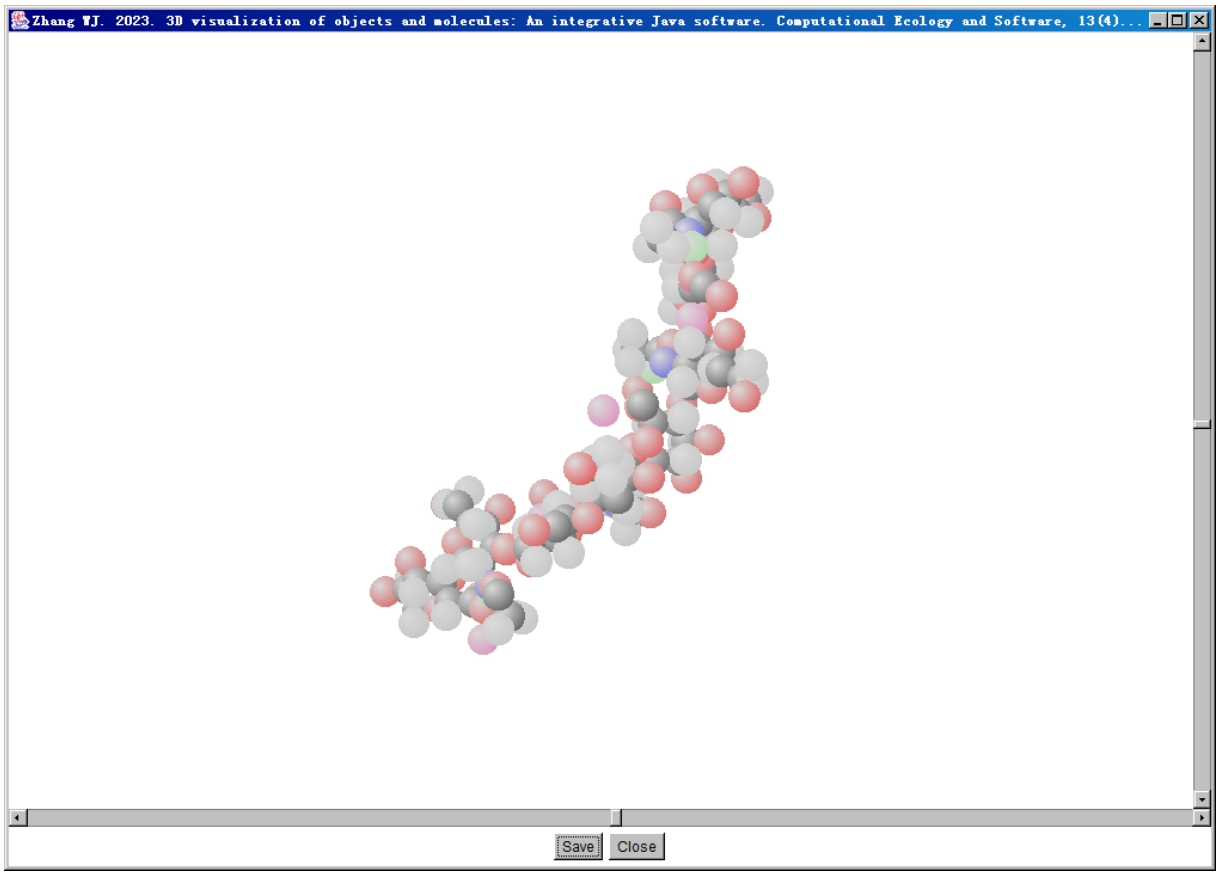

Fig. 4 The 3D graphics of hyaluronic acid.

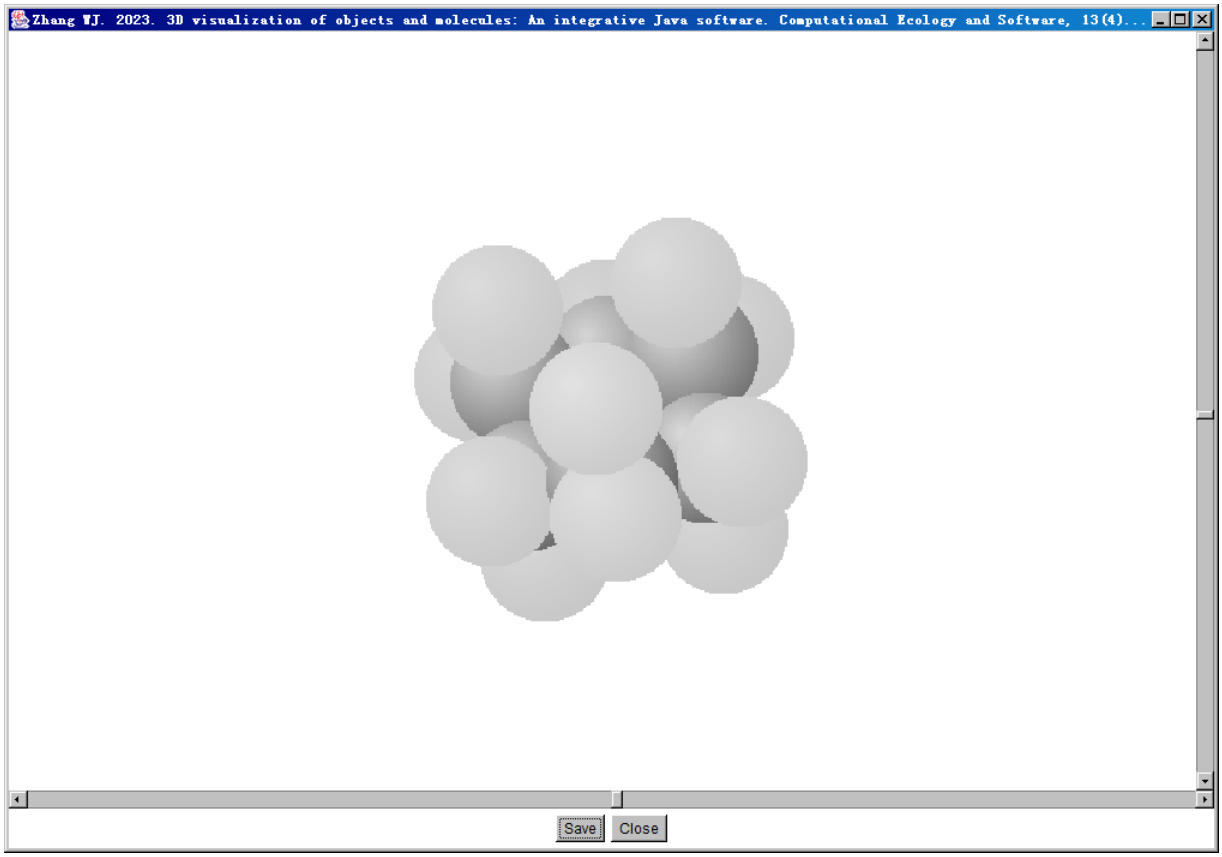

Fig. 5 The 3D graphics of cyclohexane.

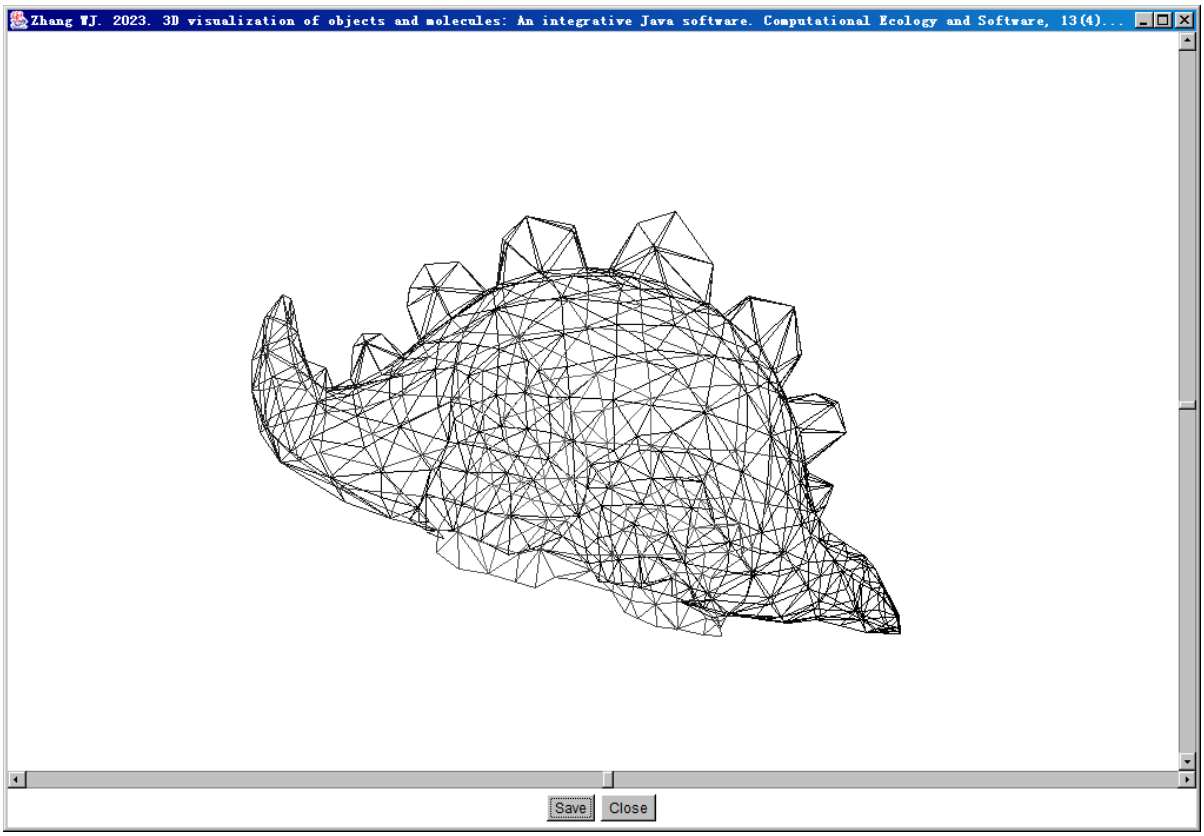

Fig. 6 The 3D graphics of a dinasaur.

Fig. 7 The 3D graphics of a helicopter.

References

Narad P, Upadhyaya KC, Som A. 2017. Reconstruction, visualization and explorative analysis of human pluripotency network. Network Biology, 7(3): 57-75

Zhang WJ. 2011a. A Java algorithm for non-parametric statistic comparison of network structure. Network Biology, 1(2): 130-133

Zhang WJ. 2011b. A Java program for non-parametric statistic comparison of community structure. Computational Ecology and Software, 1(3): 183-185

Zhang WJ. 2011c. A Java program to test homogeneity of samples and examine sampling completeness. Network Biology, 1(2): 127-129

Zhang WJ. 2007. Computer inference of network of ecological interactions from sampling data. Environmental Monitoring and Assessment, 124: 253–261

Zhang WJ. 2012a. A Java software for drawing graphs. Network Biology, 2(1): 38-44

Zhang WJ. 2012b. Computational Ecology: Graphs, Networks and Agent-based Modeling. World Scientific, Singapore

Zhang WJ. 2016. Selforganizology: The Science of Self-Organization. World Scientific, Singapore

Zhang WJ. 2018. Fundamentals of Network Biology. World Scientific Europe, London, UK

Zhang WJ. 2021. A web tool for generating user-interface interactive networks. Network Biology, 11(4): 247-262

Zhang WJ. 2024a. A Matlab software for visualizing user-interface interactive networks. Network Biology, 14(1): 13-19

Zhang WJ. 2024b. A standalone executable software for network visualization. Network Pharmacology, 9(1-2): 1-10

Zhang WJ. 2024c. An executable Java software for visualizing networks. Network Biology, 14(1): 1-12

Zhang WJ. 2024d. netGen 3.0: The executable Java software for network visualization. Selforganizology, 11(1-2): 1-27

Zhang WJ, Qi YH, Zhang ZG. 2015. A cellular automaton for population diffusion in the homogeneous rectangular area. Selforganizology, 2(1): 13-17

Zhang WJ, Zheng H. 2012. A program for statistic test of community evenness. Computational Ecology and Software, 2(1): 80-82