「ソフトコンピューティング」補助資料

## 1．Java によるアニメーション描画とデータファイル保存

Java 言語を用いた物理シミュレーションを作成において，アニメーションを行う方法はいくつか存在する．ここでは比較的シンプルなプログラム（ボタンやパネルといったイベントは一切何もない）を示す．

(Simulation.java)

Simulation.java では 2 つの図形（□と○）がそれぞれ Sin 軌道，Cos 軌道を描き，それらを互いに直線で結ぶというアニメーションである．

このプログラムの中には，

- 図形の描画
- スレッドを用いたルーチン処理(●ミリ秒ごとにイベントを繰り返し起こす)
- データのファイル出力
- 数学関数の利用

といった物理シミュレーション作成に必要な事項が盛り込まれている．

各自，Simulation.java のプログラムコードを解読し，実際の倒立振り子の制御シミュレータ作成の参考にすること．無論，ここに書いてあることは初心者に分かりやすく記述したものなので JAVA プログラミング上級者は独自のプログラムで作成してもらっても構わない．

(プログラム中の/////★★★ここから★★★/////の部分は理解できなければ読み飛ばしてもかまわない部分を示す)

```
////////////////////////////////////////////////////////////////
//
// Java 用物理シミュレーションプログラム: Simulation.java
//
////////////////////////////////////////////////////////////////

import javax.swing.*;
import java.awt.event.*;
import java.awt.*;
import java.io.*;

public class Simulation extends JFrame implements Runnable {

        ////変数の定義

        //物理変数の定義
        int t; //時間 t
        int yc,yr; //円・四角形の位置(Y 座標)

        //ファイル出力ストリーム
        FileOutputStream fout;
        PrintWriter fop;

        //スレッド変数
        Thread thr;

        /////★★★ここから★★★/////
        public static void main(String[] args){

                //シミュレーションの起動
                Simulation wnd = new Simulation();
```

```
		//終了処理
		wnd.addWindowListener(
			new WindowAdapter(){
				public void windowClosing(WindowEvent e){System.exit(0);}
			}
		);

		//実際に表示する
		wnd.setVisible(true);
	}
	/////★★★ここまで★★★/////

	//シミュレーションの実体
	public Simulation(){

		//Window タイトルの設定
		setTitle("Simulation");

		//Window サイズの決定
		setBounds( 0, 0, 600, 500);

		if (thr == null){

			//ファイル出力ストリームの初期化
			try{
				fout = new FileOutputStream("result.txt");
				fop = new PrintWriter(fout);
			}catch(FileNotFoundException e){
				System.out.println("Not Found"+e);
			}catch(IOException e){
				System.out.println("Error"+e);
			}

			//物理変数の初期化
			t=0;
			yc=100;
			yr=300;

			//スレッド（メインルーチン）の生成と開始
			thr = new Thread(this);
			thr.start();
		}
	}

	public void run(){

		//メインルーチン
		while(true){

			move();//挙動メソッド

			repaint();//描画(paint()の呼び出し)

			t++;

			//終了条件
			if(t>200){
				fop.close();
				System.exit(0);
			}

			try{
				thr.sleep(50); // 50ミリ秒停止
			}catch( Exception e){
```

```
                                        System.out.println("スリープ中に割り込みが起きました");
                                }
                        }
                }
                //50ミリ秒ごとに呼び出されるメソッド
                public void move(){

                        //ここに物理空間における物体の挙動を記述
                        yc=100+(int)(100.0*Math.sin(t*3.141592/12));
                        yr=300+(int)(100.0*Math.cos(t*3.141592/12));

                        //ファイルへ出力
                        fop.println(t+","+yc+","+yr);
                }

                //描画メソッド
                public void paint(Graphics g){

                        g.setColor( Color.white ); //色を白に設定
                        g.fillRect(0,0,600,500); //塗りつぶり四角形を描画(背景の描画)

                        g.setColor( Color.red ); //色を赤に設定
                        g.drawRect(t,yc,50,50); //四角形の描画
                        g.drawOval(t,yr,50,50); //円の描画

                        g.setColor( Color.black ); //色を黒に設定
                        g.drawLine(t+25,yc+25,t+25,yr+25); //円と四角形を結ぶ直線の描画

                        g.drawString("t:"+String.valueOf(t),100,100); //テキスト(座標値)の描画
        }
}
```

■ 簡単な説明

1) public static void main(String[] args){・・・}は main 関数である．ここでは，Simulation オブジェクトそのもの生成を行い，ウィンドウを生成する．基本的に修正を加えなくてもよい部分である．
2) public Simulation(){・・・}は Simulation クラスのコンストラクタであり，この関数内でウィンドウタイトル・サイズの決定及び，シミュレーションのメインルーチンとなるスレッド生成・開始，変数の初期化などの初期化処理を行う部分である．
3) public void run(){ while(true){・・・} }はシミュレーションにおけるメインルーチンである．ここでは t という変数が単位時間として定義されている．基本的な流れは move()で物理変数の値を計算・修正することで挙動を記述し，repaint()で paint()を呼び出すことでウィンドウに物体を描画する．そのを t を1増加させる．
   try{ }内の thr.sleep()はスレッドの停止時間を表し，この値を小さくするほど時間の流れが速くなる．基本的には無限ループ while(true){・・・}になっているため終了条件を if で記述し System.exit(0); で終了する形となる．
4) paint()は描画メソッドである．Graphics クラスのオブジェクトを引数として渡されるのでこれを使って描画メソッドを呼び出すこととなる．各メソッドは各自で調べること．
5) ファイルの出力は Simulation()内で，ファイル出力ストリームをオープン(result.txt)し，move()内で各値をファイルに出力し，run()の終了条件部でファイル出力ストリームをクローズするといった流れになっている．