```
/*
COMPLETENESS DISCLAIMER: The source code for all file, graphic, string, and memory primitives
such as FileAddLine, Gcircle, StrDelimit, and MemClear used herein is not included but could easily be
created for the platform for which this source code will be compiled
*/

/*---------------------------------------------------------------------------------------------------------------------- */
/*                                                                                                                      */
/*                                                                                                                      */
/*                                        GALACTIC BANDING ROUTINES                                                     */
/*                                                                                                                      */
/*                                                                                                                      */
/*---------------------------------------------------------------------------------------------------------------------- */
/* (c) Copyright Terence Witt 2007, All Rights Reserved */
/* NULLPHYSICS.COM */
/* GALACTIC BANDING */

/* master includes */
#include "witt.h"
#include "resource.h"
#include "zproject.h"

#define MAXRING                 1000

double  diskangle[MAXRING], diskanglestep[MAXRING], bandangle[MAXRING], bandanglestep[MAXRING];
BOOL    diskon, bandon, graphon, dualbands, fulldisk, galacticprofile, statuson;

/* initialize the banding calculation */
void InitGalacticBanding(double rotationratio, double bandcentcorr, double deltar, double diskconst,
        long diskend, double pitch, BOOL redraw)
{
        char    af[200];
        double  r, diskstepmax=(2.0*PI), pitchcorr=1.0, valu1, valu2, xx, fxvalu;
        long    n;

        /* each step is 250 ly, deltar is 500 ly */
        if(redraw)
        {
                MemClear(&diskangle, sizeof(diskangle));
                MemClear(&diskanglestep, sizeof(diskanglestep));
                MemClear(&bandangle, sizeof(bandangle));
                MemClear(&bandanglestep, sizeof(bandanglestep));
                Gframe(MESSAGEFRAME);
                Gframe(VIDEOFRAME);

                /* adjust pitch.  Corr factors are (Sa 6 deg) - 1.5, (Sb 12 deg) - 0.75, (Sc 18 deg) - 0.49
                (Milky Way Sb, 10 deg) - 0.9 */
                pitchcorr = 1.0/(diskstepmax*tan(pitch*PI/180.0));
                diskstepmax *= pitchcorr;

                /* initialize rigid and constant speed region */
                /* angular velocity for all disk radii is loaded into the diskanglestep step array */
                /* angular velocity for all disk bands is loaded into the bandanglestep step array */
                for(n=1; n<MAXRING; n++)
                {
                        r = deltar*n;

                        /* real world galactic rotation profile */
                        if(galacticprofile)
                        {
                                /* galactic velocity profile for MB -20, fxvalu is in km/s */
                                xx = r*500.0;
```

```
                        valu1 = 200.0; valu2 = 5000.0;
                        fxvalu = valu1*(1.0 - exp(-xx/valu2));
                        /* add linear increase with radius component */
                        if(xx>20000.0)
                        {
                                fxvalu += 0.0001*(xx-20000.0);
                        }
                        /* add sinusoidal component */
                        if(xx>15000.0)

                        {
                                fxvalu += 1.3*sin((valu1/200.0)*0.0003*(xx-15000));
                        }
                        /* convert km/s to diskanglestep */
                        diskanglestep[n] = (fxvalu/200.0)*diskstepmax/r;
                        if(n<diskconst)
                        {
                                /* this is assuming a flat percentage of disk rotation,
                                which is probably incorrect */
                                bandanglestep[n] = rotationratio*diskstepmax/(deltar*diskconst);
                        }else{
                                /* this is assuming a flat percentage of disk rotation,
                                which is probably incorrect */
                                bandanglestep[n] = rotationratio*diskstepmax/r;
                        }
                }

                /* simplified galactic rotation profile produces the same bands as real profile */
                if(!galacticprofile)
                {
                        if(n<diskconst)
                        {
                                diskanglestep[n] = diskstepmax/(deltar*diskconst);
                                /* this is assuming a flat percentage of disk rotation,
                                which is probably incorrect */
                                bandanglestep[n] = rotationratio*diskstepmax/(deltar*diskconst);
                        }else{
                                diskanglestep[n] = diskstepmax/r;
                                /* this is assuming a flat percentage of disk rotation,
                                which is probably incorrect */
                                bandanglestep[n] = rotationratio*diskstepmax/r;
                        }
                }
                bandangle[n] = -999;
        }

        /* current enters a fixed location */
        bandangle[diskend-1] = 0;
}

Gbold(1);
Gjust(TA_LEFT);
/* show pitch and other settings */
if(statuson)
{
        sprintf(af, "Disp:%ld  Profile:%ld  Bandcor:%3.1f  Centcor:%3.2f ",
                graphon, galacticprofile, rotationratio, bandcentcorr);
        Gtext(VIDEOFRAME, COLORWHITE, COLORBLACK, 10, 10, 36, af, TRUE);
        sprintf(af, "Pitch: %4.2f ^o", pitch);
        Gtext(VIDEOFRAME, COLORWHITE, COLORBLACK, 10, 55, 36, af, FALSE);
}else{
        Gtext(VIDEOFRAME, COLORWHITE, COLORBLACK, 10, 15, 36, "Radius: 80 Kly", FALSE);
```

```
                sprintf(af, "Pitch:   %4.2f ^o", pitch);
                Gtext(VIDEOFRAME, COLORWHITE, COLORBLACK, 10, 55, 36, af, FALSE);
        }
        Gjust(TA_CENTER);
        Gbold(1);
        Gtextframe(MESSAGEFRAME, COLORBLUE, "", "^");
        Gbold(0);
}

/*------------------------------------------------------------------------------------------------------------------------ */
/*                                                                                                                        */
/*                                    GALACTIC BANDING SIMULATION                                                         */
/*                                                                                                                        */
/*------------------------------------------------------------------------------------------------------------------------ */
/* galactic banding simulation */
void GalacticBanding(void)
{
        double  x=0, y=0, r=0, pitch=0.0, rotationratio=0.0, bandcentcorr=0.0;
        double  theta, xold, yold, valu, testbeg, testend, testcorr, gcorr=1.0;
        long    diskbeg=2, diskconst=20, diskrim=321, diskend=321, deltar=2, n;
        long    xcent=MaxViewX/2, ycent=MaxViewY/2, bandcount=0;

        /* galaxy is scaled to 4 times the milky way, each step is 250 ly, deltar is 500 ly */
        diskon = FALSE;
        bandon = FALSE;
        graphon = FALSE;
        dualbands = TRUE;
        fulldisk = FALSE;
        galacticprofile = FALSE;
        statuson = FALSE;

        /* For rotating spiral pattern. Rotation ratio is ratio of pattern speed to disk speed */
        rotationratio = 0.5;

        /* Used to match the speed of central bar with pattern speed at galactic rim.
        Only needed for barred regions */
        bandcentcorr = -0.45;

        /* to fit screen */
        gcorr = 0.8;

        /* set designated galactic pitch angle */
        pitch = 10.0;

        Gwidth(1);

        InitGalacticBanding(rotationratio, bandcentcorr, deltar, diskconst, diskend, pitch, TRUE);


        for(;;)
        {
                /* simulation on ------------------------------------------------------------------------ */
                if(graphon)
                {
                        /* erase prior */
                        for(n=diskbeg; n<diskrim; n++)
                        {
                                r = deltar*n;
                                /* show disk material movement if on */
                                if(diskon)
                                {
                                        x = xcent + (gcorr*r*sin(diskangle[n]));
```

```
                        y = ycent + (gcorr*r*cos(diskangle[n]));
                        Gcircle(VIDEOFRAME, COLORBLACK, (long)x, (long)y, 3);
                        if(dualbands)
                        {
                                x = xcent + (gcorr*r*sin(diskangle[n] + PI));
                                y = ycent + (gcorr*r*cos(diskangle[n] + PI));
                                Gcircle(VIDEOFRAME, COLORBLACK, (long)x, (long)y, 3);
                        }
                }
                if(bandon)
                {
                        if(bandangle[n]>-999.0)
                        {
                                x = xcent + (gcorr*r*sin(bandangle[n]));
                                y = ycent + (gcorr*r*cos(bandangle[n]));
                                Gcircle(VIDEOFRAME, COLORBLACK, (long)x, (long)y, 3);
                                if(dualbands)
                                {
                                        x = xcent + (gcorr*r*sin(bandangle[n] + PI));
                                        y = ycent + (gcorr*r*cos(bandangle[n] + PI));
                                        Gcircle(VIDEOFRAME, COLORBLACK, (long)x, (long)y, 3);
                                }
                        }
                }
}

/* rotate the disk at rigid inner region and constant speed outer region */
for(n=diskbeg; n<diskend; n++)
{
        diskangle[n] += diskanglestep[n];
}
for(n=diskbeg; n<diskend; n++)
{
        bandangle[n] += bandanglestep[n];
}

/* vortical inflow ------------------------------------------------------------- */
if(!(bandcount++%deltar))
{
        for(n=diskbeg; n<diskend; n++)
        {
                /* move inward at a certain constant rate, blown around by the disk */
                if(bandangle[n+1]>-999.0)
                {
                        if(n>diskconst)
                        {
                                /* in the constant speed region, a band's position is
                                the product of its former position and local disk

                                movement, scaled by the distance between the rings */
                                bandangle[n] = bandangle[n+1] +
                                        deltar*diskanglestep[n+1];
                        }else{
                                /* in the rigid body region,
                                current moves straight inward */
                                bandangle[n] = bandangle[n+1] +
                                        bandcentcorr*deltar*diskanglestep[n+1];
                        }
                }
        }
}
```

```
	/* draw ---------------------------------------------------------------------- */
	for(n=diskbeg; n<diskrim; n++)
	{
		r = deltar*n;
		if(fulldisk)
		{
			Gfill(0);
			Gcircle(VIDEOFRAME, COLORGREEN, xcent, ycent, (long)r);
		}
		Gfill(1);
		if(diskon)
		{
			/* disk material movement */
			x = xcent + (gcorr*r*sin(diskangle[n]));
			y = ycent + (gcorr*r*cos(diskangle[n]));
			Gcircle(VIDEOFRAME, COLORGREEN, (long)x, (long)y, 3);
			if(dualbands)
			{
				x = xcent + (gcorr*r*sin(diskangle[n] + PI));
				y = ycent + (gcorr*r*cos(diskangle[n] + PI));
				Gcircle(VIDEOFRAME, COLORGREEN, (long)x, (long)y, 3);
			}
		}
		if(bandon)
		{
			if(bandangle[n]>-999.0)
			{
				/* band stays fixed but rotates */
				x = xcent + (gcorr*r*sin(bandangle[n]));
				y = ycent + (gcorr*r*cos(bandangle[n]));
				Gcircle(VIDEOFRAME, COLORSKY, (long)x, (long)y, 3);
				if(dualbands)
				{
					x = xcent + (gcorr*r*sin(bandangle[n] + PI));
					y = ycent + (gcorr*r*cos(bandangle[n] + PI));
					Gcircle(VIDEOFRAME, COLORSKY, (long)x, (long)y, 3);
				}
			}
		}
	}
	/* let refresh catch up */
	Pause(0.02);
}

/* keyboard entry ------------------------------------------------------------------------ */
/* turn galactic profile on and off */
if(KeyDown('G'))
{
	if(galacticprofile)
	{
		galacticprofile = FALSE;
	}else{
		galacticprofile = TRUE;
	}
	InitGalacticBanding(rotationratio, bandcentcorr, deltar,
		diskconst, diskend, pitch, TRUE);
	while(KeyDown('G'));
}

/* turn disk material movement display on and off */
if(KeyDown('D'))
{
```

```
		if(diskon)
		{
			diskon = FALSE;
		}else{
			diskon = TRUE;
		}
		InitGalacticBanding(rotationratio, bandcentcorr, deltar,
			diskconst, diskend, pitch, TRUE);
		while(KeyDown('D'));
}

/* turn band movement display on and off */
if(KeyDown('B'))
{
		if(bandon)
		{
			bandon = FALSE;
		}else{
			bandon = TRUE;
		}
		InitGalacticBanding(rotationratio, bandcentcorr, deltar,
			diskconst, diskend, pitch, TRUE);
		while(KeyDown('B'));
}

/* adjust central core co-rotation */
if(KeyDown(VK_UP))
{
		bandcentcorr += 0.05;
		InitGalacticBanding(rotationratio, bandcentcorr, deltar,
			diskconst, diskend, pitch, FALSE);
		while(KeyDown(VK_UP));
}
if(KeyDown(VK_DOWN))
{
		bandcentcorr -= 0.05;
		if(fabs(bandcentcorr)<0.02)
		{
			bandcentcorr = 0.0;
		}
		InitGalacticBanding(rotationratio, bandcentcorr, deltar,
			diskconst, diskend, pitch, FALSE);
		while(KeyDown(VK_DOWN));
}

/* adjust band co-rotation */
if(KeyDown(VK_RIGHT))
{
		rotationratio += 0.1;
		InitGalacticBanding(rotationratio, bandcentcorr, deltar,
			diskconst, diskend, pitch, TRUE);
		while(KeyDown(VK_RIGHT));
}
if(KeyDown(VK_LEFT))
{
		rotationratio -= 0.1;
		InitGalacticBanding(rotationratio, bandcentcorr, deltar,
			diskconst, diskend, pitch, TRUE);
		while(KeyDown(VK_LEFT));
}

/* draw a dual exponential spiral curve for comparison dr/dtheta = R */
```

```
		if(KeyDown('T'))
		{

			Gwidth(3);
			/* do at least 9 turns */
			testbeg = (0.5*PI) + 9.0*(2.0*PI);
			testend = 2.0*PI;
			valu = tan(pitch*PI/180.0);
			testcorr = (deltar*(diskrim-1))/exp(valu*testbeg);
			xold = xcent;
			yold = ycent + gcorr*testcorr*exp(valu*testbeg);
			for(theta=testbeg; theta>=testend; theta-=0.01)
			{
				r = gcorr*testcorr*exp(valu*theta);
				x = xcent + r*cos(theta);
				y = ycent + r*sin(theta);
				Gline(VIDEOFRAME, COLORPURPLE, (long)xold, (long)yold, (long)x, (long)y);
				xold = x;
				yold = y;
			}
			Gwidth(1);
			while(KeyDown('T'));
		}

		/* pause motion */
		if(KeyDown(' '))
		{
			if(graphon)
			{
				graphon = FALSE;
			}else{
				graphon = TRUE;
			}
			InitGalacticBanding(rotationratio, bandcentcorr, deltar,
				diskconst, diskend, pitch, FALSE);
			while(KeyDown(' '));
		}

		/* grab bitmap */
		if(KeyDown(VK_F5))
		{
			strcpy(note_text, "FULL");
			CaptureBitmap();
			while(KeyDown(VK_F5));
		}

		/* exit */
		if(KeyDown(VK_ESCAPE))

		{
			break;
		}
	}
}
```