



# Урок №3

Анимация. Движение по  
криволинейной траектории

# Функция **GetTickCount**

**DWORD function GetTickCount ( ) ;**

Функция Windows API, возвращающая количество миллисекунд с момента старта Windows.

**DWORD** – беззнаковый целочисленный тип, размером двойное машинное слово.

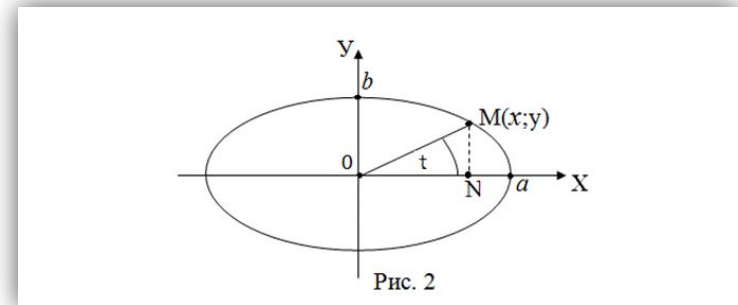
# Равномерное движение по кривой, заданной в параметрическом виде

Самая простая криволинейная траектория - это окружность или эллипс. Окружность является частным случаем эллипса.

Параметрическое уравнение эллипса в прямоугольной системе координат имеет вид:

$$\begin{cases} x = a \cos t \\ y = b \sin t \end{cases} \quad 0 \leq t \leq 2\pi,$$

где  $t$  — параметр.



Параметр  $t$  является углом между положительным направлением оси абсцисс и радиус-вектором данной точки.

**Перед нами стоит задача – изобразить круг, движущийся по эллиптической орбите с некоторой угловой скоростью.**

У нас уже имеется функция рисования круга:

```
void DrawRound(double x, double y, double radius, double
    r, double g, double b)
{
    // передаются координаты центра, радиус и цвет круга
    int n = 50; // количество вершин полигона
    glBegin(GL_POLYGON);
        glColor3d(r, g, b);
        for (int i = 0; i < n; i++)
            glVertex2d(x + radius*cos(i * 2 * 3.14159 / n),
                y + radius*sin(i * 2 * 3.14159 / n));
    glEnd();
}
```

Используем ее для рисования движущегося круга.

Функция `moveRound()` – движение круга по эллиптической орбите:

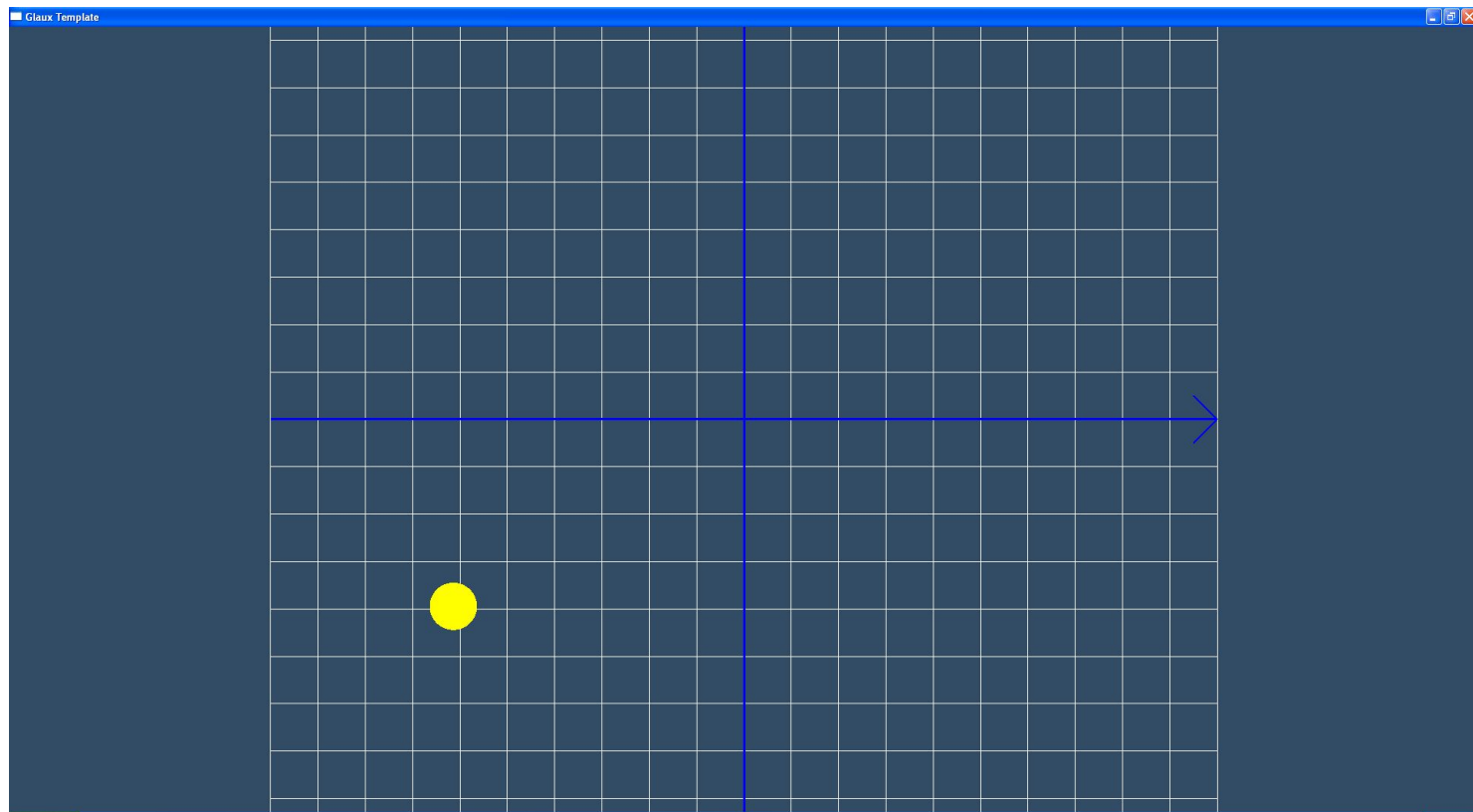
```
//добавьте глобальные переменные
const float PI=3.14159f;
float v1 = 90;//скорость в градусах
float startTime = 0.0f;//время запуска программы
void moveRound()
{
    float dt = (GetTickCount() - startTime)/1000.f;
    float x1 = 10 * cos(dt*v1*PI/180);//если PI взять
    =3, то получим(dt*v1/60)
    float y1 = 5 * sin(dt*v1*PI/180);
    DrawRound(x1,y1,0.5,1,1,0);
}
```

Отредактируйте функцию display (или Draw):

```
void CALLBACK display(void)
{
    glClear( GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT );
    moveRound(); //движущийся круг
    osi(10); //оси координат
    auxSwapBuffers();
}
```

Отредактируйте функцию main:

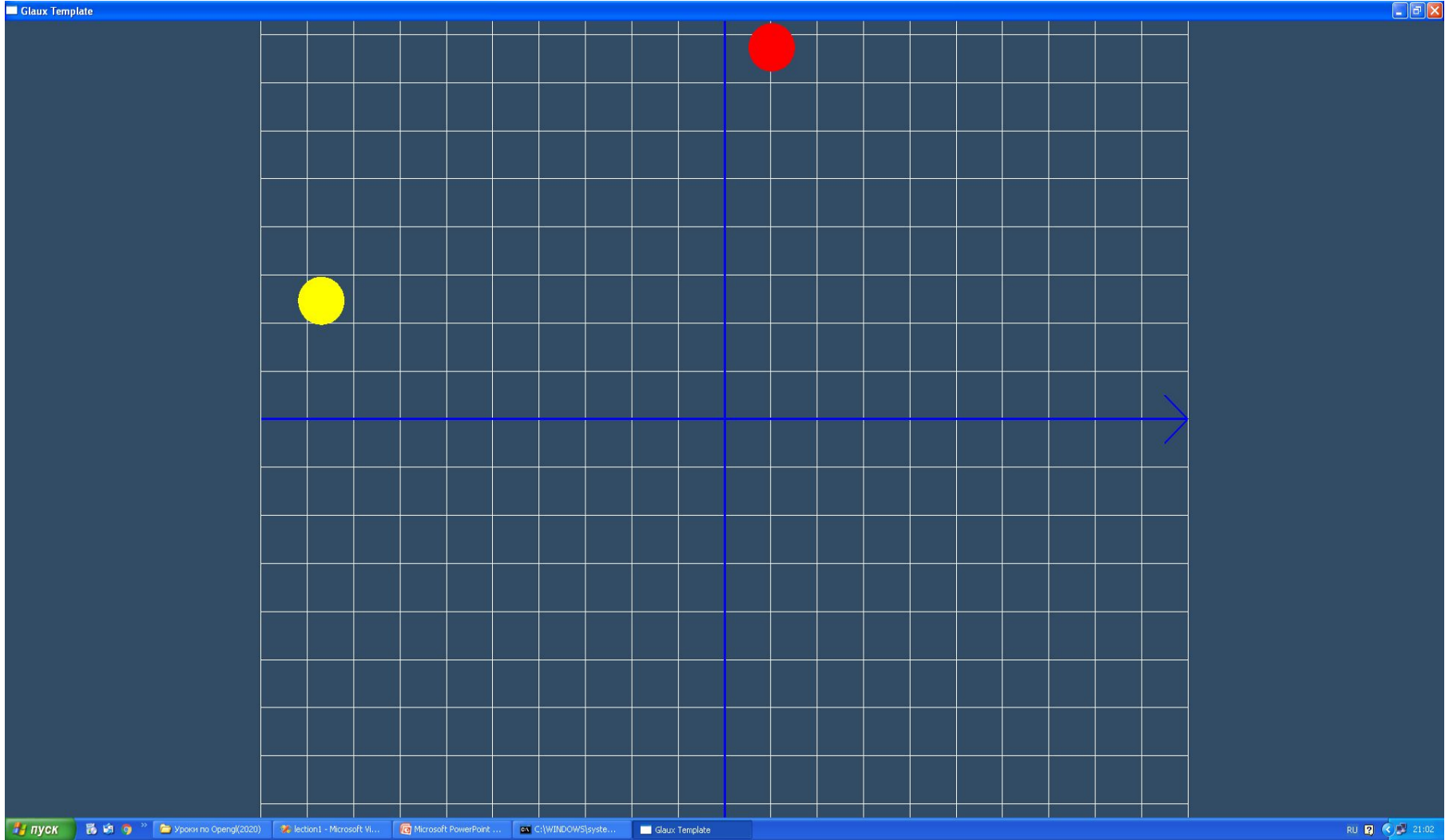
```
void main()
{
    startTime=GetTickCount();
    RunOpenGL();
}
```



Результат работы программы

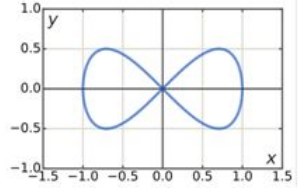
Траектория движения круга – эллипс с большой полуосью =10, малой = 5, угловой скоростью  $v_1=90$ .

Самостоятельно добавьте еще один круг, траектория движения которого, например, эллипс с большой полуосью = 4, малой = 8, угловой скоростью = 45.





# Некоторые кривые, формулы которых можно использовать для задания траектории:

Уравнения			
1	Синусоида	$y = a \cdot \sin(k \cdot x) + b$	
2	Косинусоида	$y = a \cdot \cos(k \cdot x) + b$	
3	Парабола	$y = a \cdot x^2 + b \cdot x + c$	
4	Свое уравнение		
Параметрические			
4	Парабола	$\rho = a / (1 + \cos(\varphi))$	
5	Спираль Архимеда	$\rho = a \cdot \varphi, a \neq 0$	
6	Логарифмическая спираль	$\rho = a \cdot \exp(k \cdot \varphi)$	
7	Параболическая спираль	$\rho = a \cdot \sqrt{\varphi} + l,$ $a \neq 0, l \geq 0$	
8	Гиперболическая спираль	$\rho = a / \varphi, a \neq 0$	
9	Эллипс	$x = a \cdot \cos(\varphi),$ $y = b \cdot \sin(\varphi)$	
10	Лемниската	$\rho = a \cdot \sqrt{2 \cdot \cos(2 \cdot \varphi)}$	
11	Лемниската Жероно	$x = \cos(\varphi)$ $y = \sin(\varphi) \cdot \cos(\varphi)$	

12	Улитка Паскаля	$-\rho = 2 \cdot R \cdot \cos(\varphi) + a$	
13	Кардиоида	$\rho = a \cdot (1 - \cos(\varphi)),$ $a$ -диаметр окружностей	
14	Роза Гранди	$\rho = a \cdot \sin(k \cdot \varphi), k = n/d$	
15	Астроида	$x = R \cdot \cos^3(\varphi)$ $y = R \cdot \sin^3(\varphi)$	
16	Своя параметрическая функция		

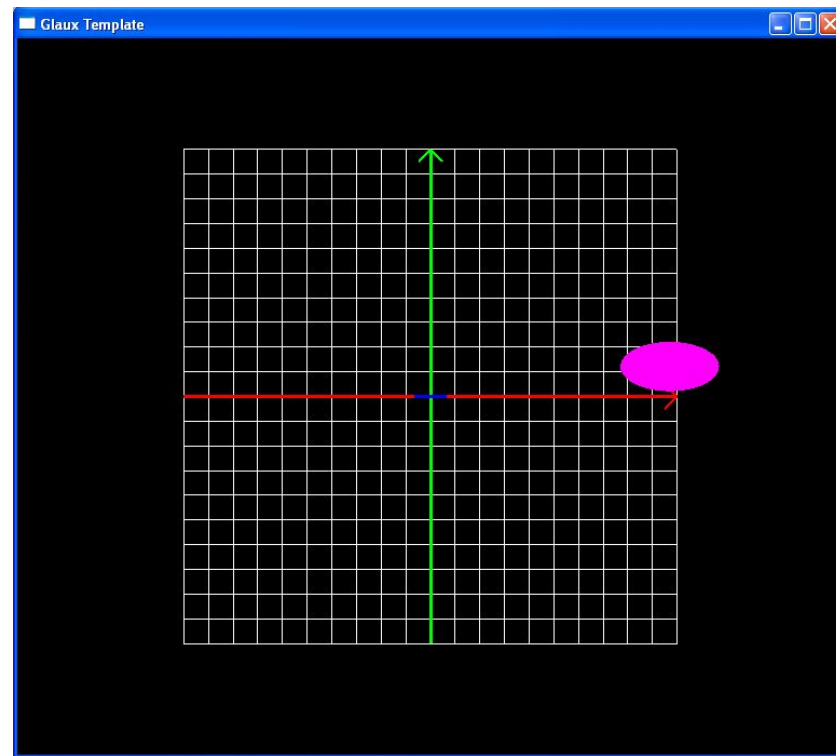
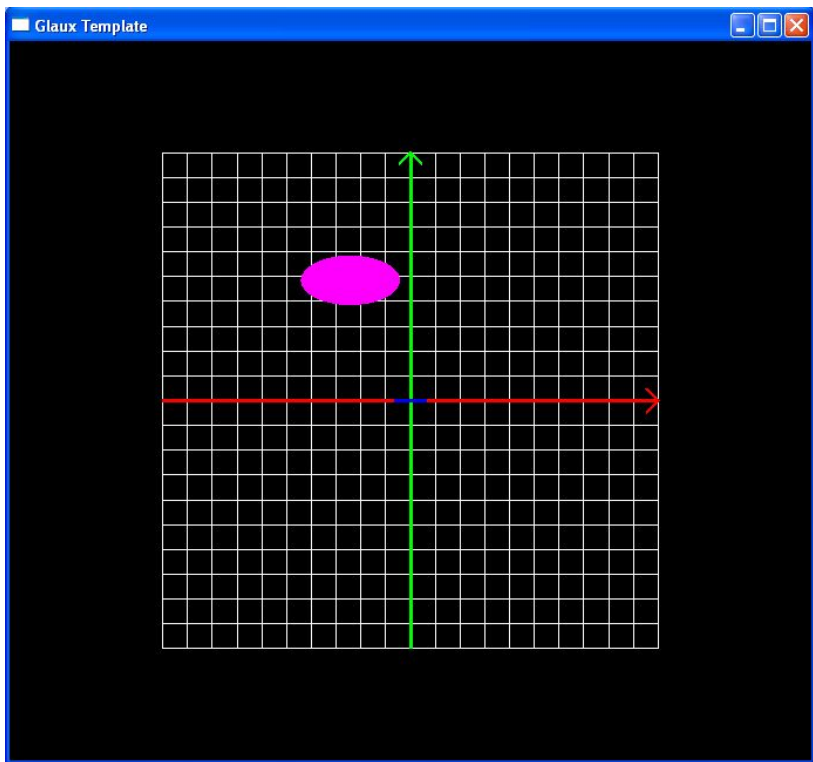
А как анимировать объект, если он отрисовывается всегда в одном месте, например, в начале координат?

Наш эллипс всегда рисуется в начале координат. Вот наша функция рисования заполненного эллипса:

```
void DrawEllipseFill(double r1, double r2, double r, double
    g, double b)
{ // передаются полуоси, цвет эллипса
  int n = 50; // количество точек полигона
  glBegin(GL_TRIANGLE_FAN);
  glColor3d(r, g, b);
  for (int i = 0; i < n; i++)
    glVertex2d(r1*cos(i*2*3.14159/n), r2*sin(i*2*3.14159/n));
  glEnd();
}
```

Очень просто! Переместим его в нужную нам точку с помощью `glTranslated`.

```
const float PI=3.14159f;
float v1 = 90;//угловая скорость
float startTime = 0.0f; //время запуска программы
//Функция рисования движущегося эллипса
void moveEllipseFill()
{ float dt = (GetTickCount() - startTime)/1000.f;
  float x1 = 10 * cos(dt*v1*PI/180); //если PI взять =3, то
    (dt*v1/60)
  float y1 = 5 * sin(dt*v1*PI/180);
  glPushMatrix();//сохранили текущую матрицу
  glTranslated(x1, y1, 0); //переместили систему в точку (x1,y1)
  DrawEllipseFill(2,1,1,0,1); //нарисовали эллипс нужного
    размера и цвета
  glPopMatrix();//вернули текущую матрицу
}
```



Траектория движения эллипса – эллиптическая с большой полуосью =10, малой = 5.

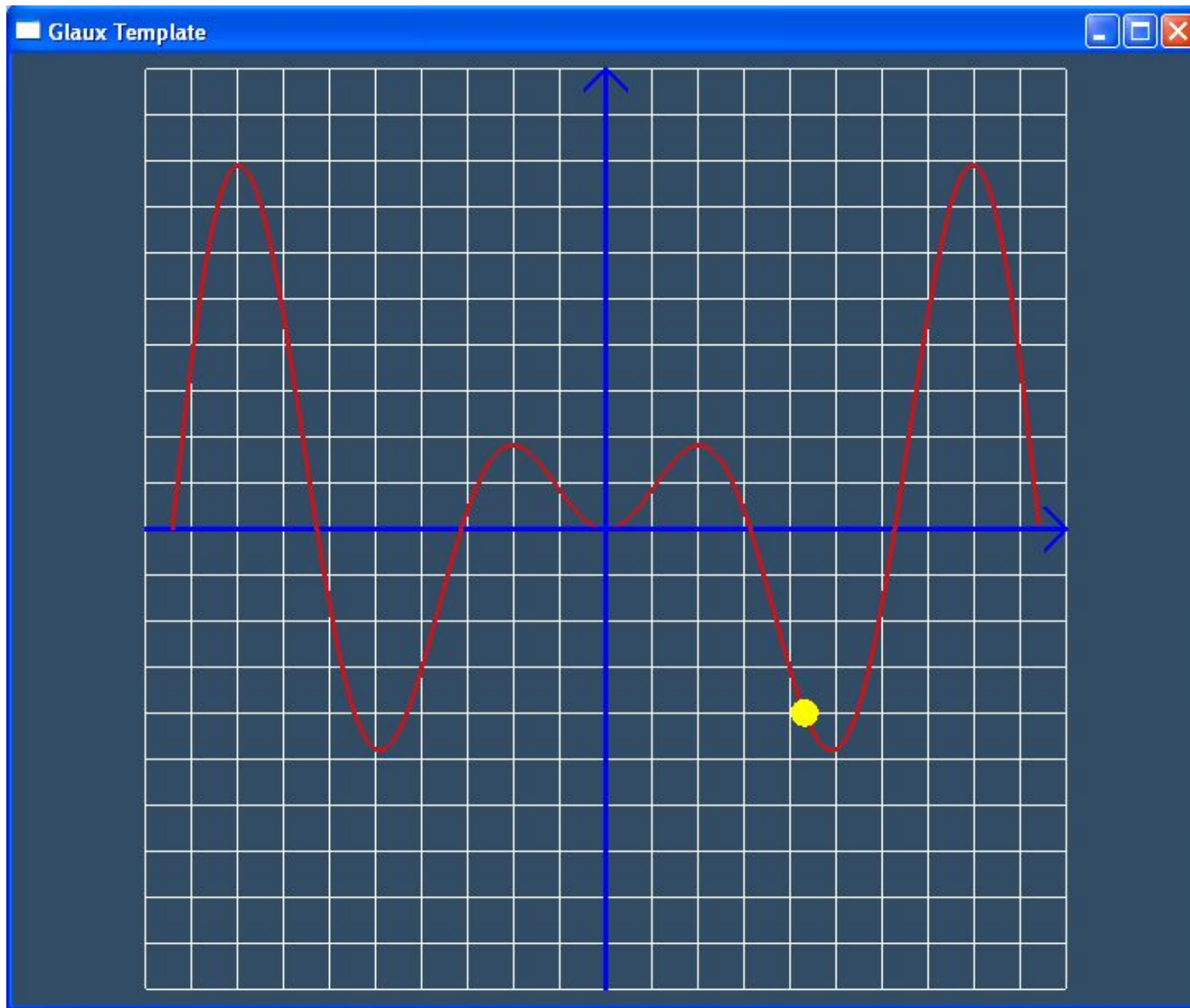
# Движение по траектории, заданной аналитической функцией $y=f(x)$

```
float xFrom=-3*PI, //начальная точка движения
      xTo=3*PI; // конечная точка движения
float v=4; //скорость ед/с
void moveRound1()
{
    float dt = (GetTickCount() - startTime)/1000.f;
    float x1=xFrom+dt*v;
    //код с if необходим, чтобы объект двигался и в обратном направлении
    if ((x1>xTo) && (xTo>xFrom) || (x1<xTo) && (xTo<xFrom))
    {
        v = -v;
        float xTemp = xTo;
        xTo = xFrom;
        xFrom = xTemp;
        startTime = GetTickCount();
    }

    float y1=x1*sin(x1); // уравнение кривой y=x*sin(x)
    DrawRound(x1, y1, 0.3, 1, 1, 0);
}
```

Добавьте вызов `moveRound1()` в функцию `display()` :

```
void CALLBACK display(void)
{
    glClear( GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT );
    moveRound1();
    osi(10);
    auxSwapBuffers();
}
```



Круг движется по траектории  $y = x \cdot \sin(x)$ , на отрезке  $[-3\pi, 3\pi]$ ,  
(график из лабораторной работы №3)