ГУАП

КАФЕДРА № 42

ОТЧЕТ
ЗАЩИЩЕН С ОЦЕНКОЙ

ПРЕПОДАВАТЕЛЬ

доцент		А. В. Аграновский
должность, уч. степень, звание	подпись, дата	инициалы, фамилия

ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ №6

Сплайновая кривая Безье

по курсу: Компьютерная Графика

РАБОТУ ВЫПОЛНИЛ

СТУДЕНТ гр. № 4321		К. А. Лебедев
	подпись, дата	инициалы, фамилия

Санкт-Петербург 2024

Цель работы

Изучение сплайновой кривой Безье, построение сплайновой кривой Безье с помощью математического пакета и/или языка программирования высокого уровня.

Формулировка задания

1. Построить график гармонических колебаний.
2. На периоде гармонических колебаний взять N точек, где N равно 4 плюс номер студента в группе.
3. По опорным точкам из пункта 2 построить кривую Безье (на том же графике, что и в пункте 1).
4. Рассчитать ошибку восстановления гармонических колебаний кривой Безье.
5. Уменьшить число точек на периоде в 2 раза и повторить предыдущие пункты.
6. Увеличить число точек на периоде в 2 раза и повторить первые 4 пункта.
7. Построить кривую Безье на основе полинома N-го порядка (где N берется из пункта 2) и рассчитать ошибку.

Теоретические положения, используемые при выполнении лабораторной работы.

Сплайны представляют собой гладкие кривые, которые строятся на основе набора опорных точек и удовлетворяют определённым критериям гладкости. Одним из наиболее распространённых типов сплайнов являются кривые Безье, которые находят применение в компьютерной графике для создания плавных изгибов. Кривая Безье определяется как комбинация полиномов Бернштейна, где каждая опорная точка влияет на форму кривой.

Для набора из четырёх опорных точек P_0, P_1, P_2, элементарная кубическая кривая Безье задаётся следующим уравнением:

$R(t)=(1-t)^3P_0+3t(1-t)^2P_1+3t^2(1-t)P^2+t^3P_3$, $0\leq t\leq 1$.

Эта кривая начинается в точке P0P0 и заканчивается в точке P3P3, при этом касаясь прямых P_0P_1 и P_2P_3. Одним из ключевых свойств кривых Безье является то, что они проходят внутри выпуклой оболочки, образованной опорными точками, а форма кривой определяется только заданным набором точек и весовыми коэффициентами.

Для построения сложных кривых используется подход, при котором опорные точки разбиваются на группы по четыре. Для каждой группы строится элементарная кубическая кривая Безье, а затем полученные сегменты соединяются. Чтобы кривая была геометрически непрерывной, необходимо, чтобы каждые три точки на границах сегментов лежали на одной прямой. В случае, если количество точек не делится на четыре, последняя точка может повторяться несколько раз.

Кривые Безье также используются для аппроксимации других функций. Например, для восстановления заданной функции, такой как гармонические колебания, строится кривая Безье на основе набора опорных точек, расположенных на интервале функции. Ошибка восстановления вычисляется как среднее абсолютное отклонение между значениями исходной функции и построенной кривой Безье.

Кроме того, для сравнения точности аппроксимации может быть использован полином степени $N-1$, где N— количество опорных точек. Полином записывается в виде:

$$P(x)=a_0+a_1x+a_2x^2+\cdots+a_{N-1}x^{N-1},$$

где коэффициенты a_i определяются решением системы линейных уравнений, составленных на основе опорных точек. Такой подход позволяет оценить точность полиномиальной аппроксимации по сравнению с использованием кривых Безье.

Листинг с кодом программы.

```
const canvas1 = document.getElementById("canvas1");
const canvas2 = document.getElementById("canvas2");
const canvas3 = document.getElementById("canvas3");
const canvas4 = document.getElementById("canvas4");

const canvases = [canvas1, canvas2, canvas3, canvas4];
const contexts = canvases.map(canvas => canvas.getContext("2d"));

// Константы

const baseN = 14; // Число опорных точек
const period = 2 * Math.PI; // Период гармонического колебания
const amplitude = 1; // Амплитуда колебания
const steps = 100; // Точность для отрисовки кривой

// Функция гармонического колебания
function harmonicFunction(t) {
    return Math.sin(t);
}

// Генерация точек
function generatePoints(N) {
    const points = [];
    const step = period / N;
    for (let i = 0; i <= N; i++) {
        const t = i * step;
        const x = (t / period) * canvas1.width;
        const y =
            canvas1.height / 2 -
```

```
            harmonicFunction(t) * (canvas1.height / 2 - 20);
        points.push({ x, y });
    }
    return points;
}

// Алгоритм Де Кастельжо для кривой Безье
function calculateBezier(points, t) {
    let tempPoints = [...points];
    while (tempPoints.length > 1) {
        const nextPoints = [];
        for (let i = 0; i < tempPoints.length - 1; i++) {
            const x = (1 - t) * tempPoints[i].x + t * tempPoints[i + 1].x;
            const y = (1 - t) * tempPoints[i].y + t * tempPoints[i + 1].y;
            nextPoints.push({ x, y });
        }
        tempPoints = nextPoints;
    }
    return tempPoints[0];
}

// Генерация кривой Безье
function bezierCurve(points, steps = 100) {
    const curve = [];
    for (let t = 0; t <= 1; t += 1 / steps) {
        curve.push(calculateBezier(points, t));
    }
    return curve;
}
```

```
// Расчет ошибки
function calculateError(original, bezier) {
    let error = 0;
    original.forEach((point, index) => {
        const closest = bezier.reduce((min, cur) =>
            Math.abs(cur.x - point.x) < Math.abs(min.x - point.x) ? cur : min
        );
        error += Math.abs(point.y - closest.y);
    });
    return error / original.length;
}

// Отрисовка гармонической функции
function drawHarmonic(ctx, color = "blue") {
    ctx.beginPath();
    for (let t = 0; t <= period; t += period / steps) {
        const x = (t / period) * canvas1.width;
        const y =
            canvas1.height / 2 -
            harmonicFunction(t) * (canvas1.height / 2 - 20);
        if (t === 0) ctx.moveTo(x, y);
        else ctx.lineTo(x, y);
    }
    ctx.strokeStyle = color;
    ctx.lineWidth = 2;
    ctx.stroke();
}

// Отрисовка кривой Безье
function drawBezier(ctx, curve, color = "red") {
```

```
    ctx.beginPath();
    ctx.moveTo(curve[0].x, curve[0].y);
    curve.forEach(point => ctx.lineTo(point.x, point.y));
    ctx.strokeStyle = color;
    ctx.lineWidth = 2;
    ctx.stroke();
}

// Отрисовка точек
function drawPoints(ctx, points, color = "black") {
    points.forEach(({ x, y }) => {
        ctx.beginPath();
        ctx.arc(x, y, 4, 0, Math.PI * 2);
        ctx.fillStyle = color;
        ctx.fill();
    });
}

// Основная функция
function main() {
    const scenarios = [
        { N: baseN, canvas: canvas1, label: `N = ${baseN}` },
        {
            N: Math.floor(baseN / 2),
            canvas: canvas2,
            label: `N = ${Math.floor(baseN / 2)}`,
        },
        { N: baseN * 2, canvas: canvas3, label: `N = ${baseN * 2}` },
        {
            N: baseN,
```

```
            canvas: canvas4,
            label: `Ошибка восстановления (N = ${baseN})`,
        },
    ];

    scenarios.forEach(({ N, canvas, label }, i) => {
        const ctx = contexts[i];
        ctx.clearRect(0, 0, canvas.width, canvas.height);
        ctx.font = "16px Arial";
        ctx.fillText(label, 10, 20);

        // Гармоническое колебание
        drawHarmonic(ctx);

        // Опорные точки
        const points = generatePoints(N);
        drawPoints(ctx, points);

        // Кривая Безье
        const bezier = bezierCurve(points);
        drawBezier(ctx, bezier, "red");

        // Ошибка
        if (i === scenarios.length - 1) {
            const error = calculateError(points, bezier);
            ctx.fillText(`Ошибка: ${error.toFixed(5)}`, 10, 40);
        }
    });
}
main();
```

Экранные формы с результатом работы программы.

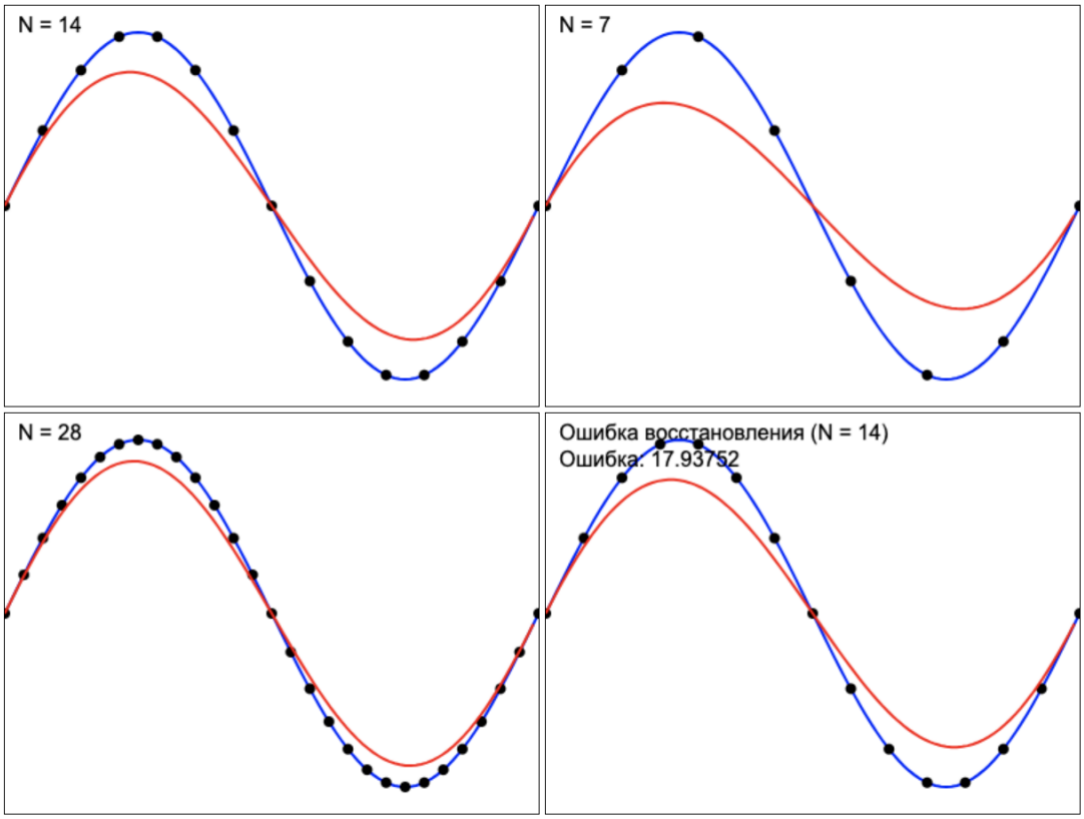

Рисунок 1 – результат работы программы

Вывод

В ходе выполнения лабораторной работы были исследованы свойства кривых Безье и их применение для аппроксимации гармонических колебаний. Основные результаты заключаются в следующем:

Построены графики гармонических колебаний и кривых Безье для разных наборов опорных точек, что позволило наглядно изучить процесс аппроксимации.

Выполнен расчет ошибок восстановления, выявлена зависимость точности аппроксимации от количества опорных точек. Было отмечено, что увеличение числа опорных точек улучшает точность аппроксимации кривой Безье, поскольку она лучше повторяет форму исходной функции.

Проведено сравнение аппроксимации кривой Безье с использованием полиномов. Установлено, что при малом количестве опорных точек полиномиальная аппроксимация демонстрирует хорошие результаты, однако при увеличении их числа точность ухудшается из-за эффекта переоснащения, характерного для полиномов высокой степени.

На основании полученных данных можно сделать вывод, что кривые Безье являются более устойчивым методом аппроксимации по сравнению с полиномами, особенно при увеличении числа опорных точек. Таким образом, использование кривых Безье предпочтительно для точного восстановления сложных функций.