Міністерство освіти і науки України

Національний університет «Запорізька політехніка»

кафедра програмних засобів

ЗВІТ

з завдання

з дисципліни «Комп'ютерна графіка та обробка зображень» на тему:

«АЛГОРИТМИ ПОБУДОВИ ПРЯМОЇ ЛІНІЇ»

Виконав:

ст. гр. КНТ-113сп Іван ЩЕДРОВСЬКИЙ

Прийняв:

Доцент Анжеліка ПАРХОМЕНКО

2025

1 Завдання до лабораторної роботи

Дано: координати точки P1 (2,4); координати точки P2 (11,7).

Визначити координати пікселів відрізка лінії, що обмежується цими точками на основі реалізації чотиризв’язного алгоритму Брезенхейма

Порівняти отримані результати з результатами виконання Задачі на слайді 4

2 Виконання лабораторної роботи

Для виконання роботи було створено програму-сайт, яка дозволяє візуалізувати результат алгоритму

Програма підтримує роботу в чотиризв’язному та восьмизв’язному режимах

Також в програмі можна змінювати позиції точок. Лівий клік мишу для першої, правий клік миші для другої

Результат виконання завдання з заданими точками P1, P2 відповідає результату з лекції

На рисунку 1 показаний режим роботи програми в восьмизв’язному режимі. На рисунку 2 в чотирьохзв’язному. На рисунку 3 показано інші точки

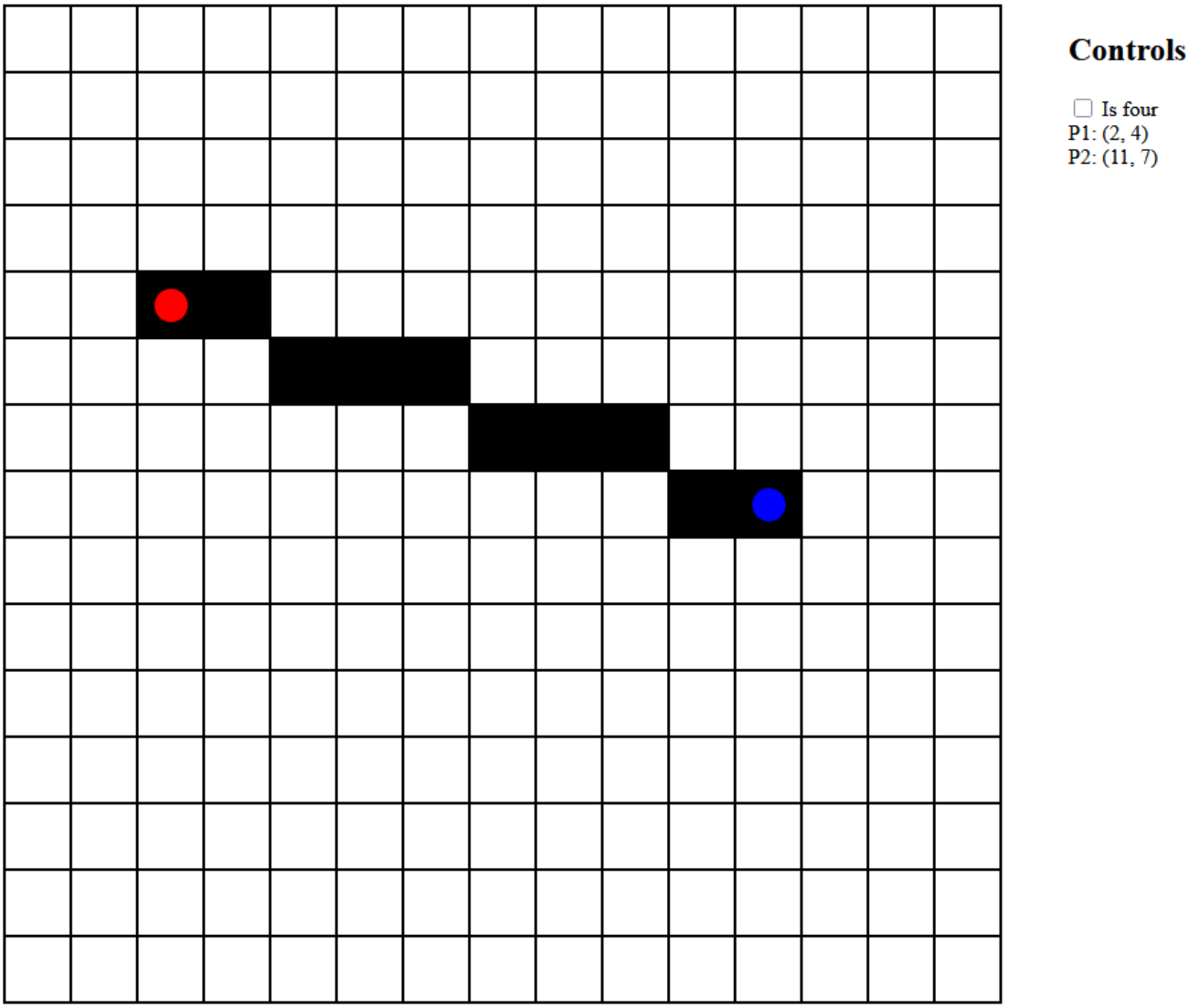

Рисунок 1 – Демонтрація роботи

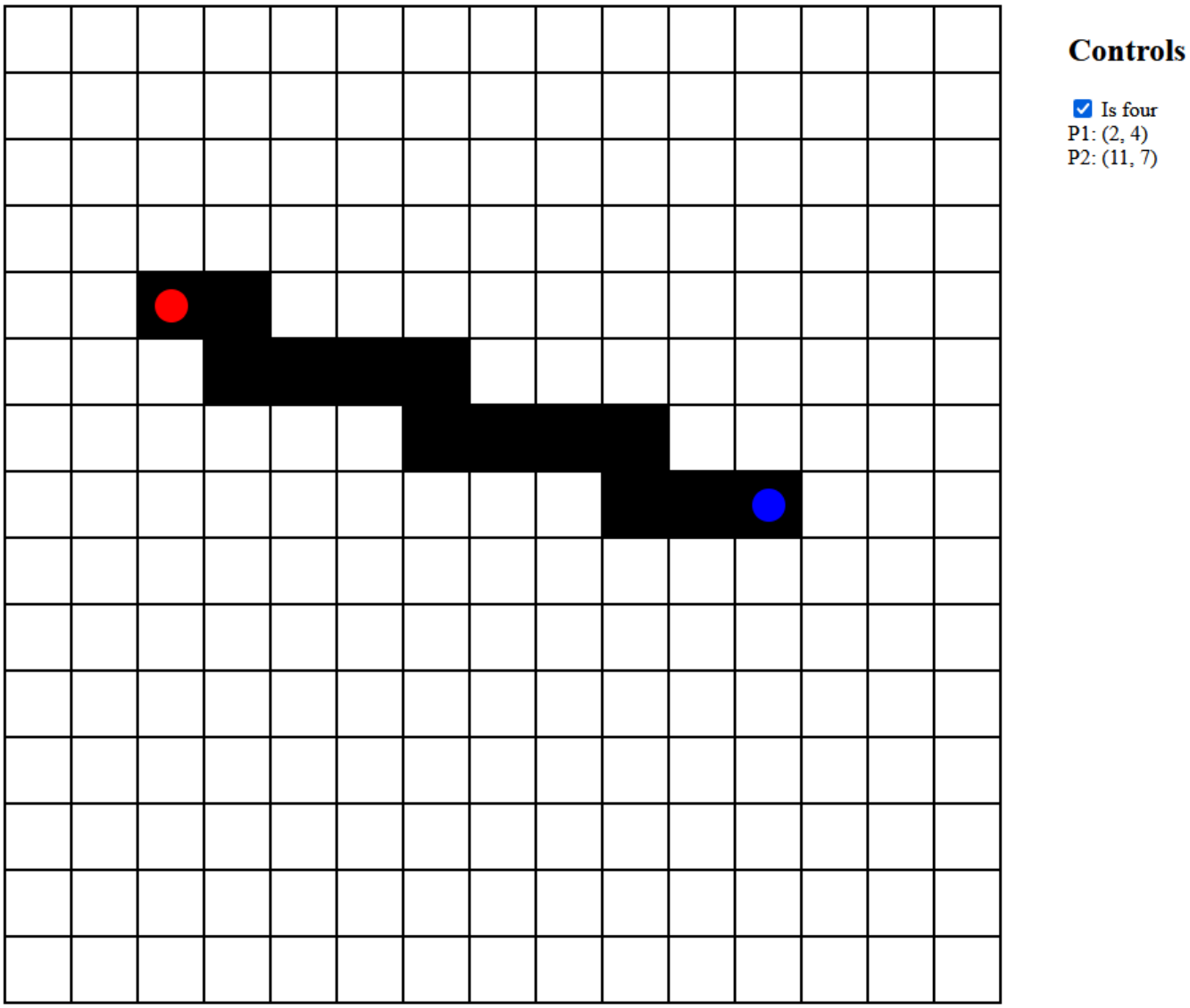

Рисунок 2 – Демонстрація роботи в чотирьохзв’язному режимі

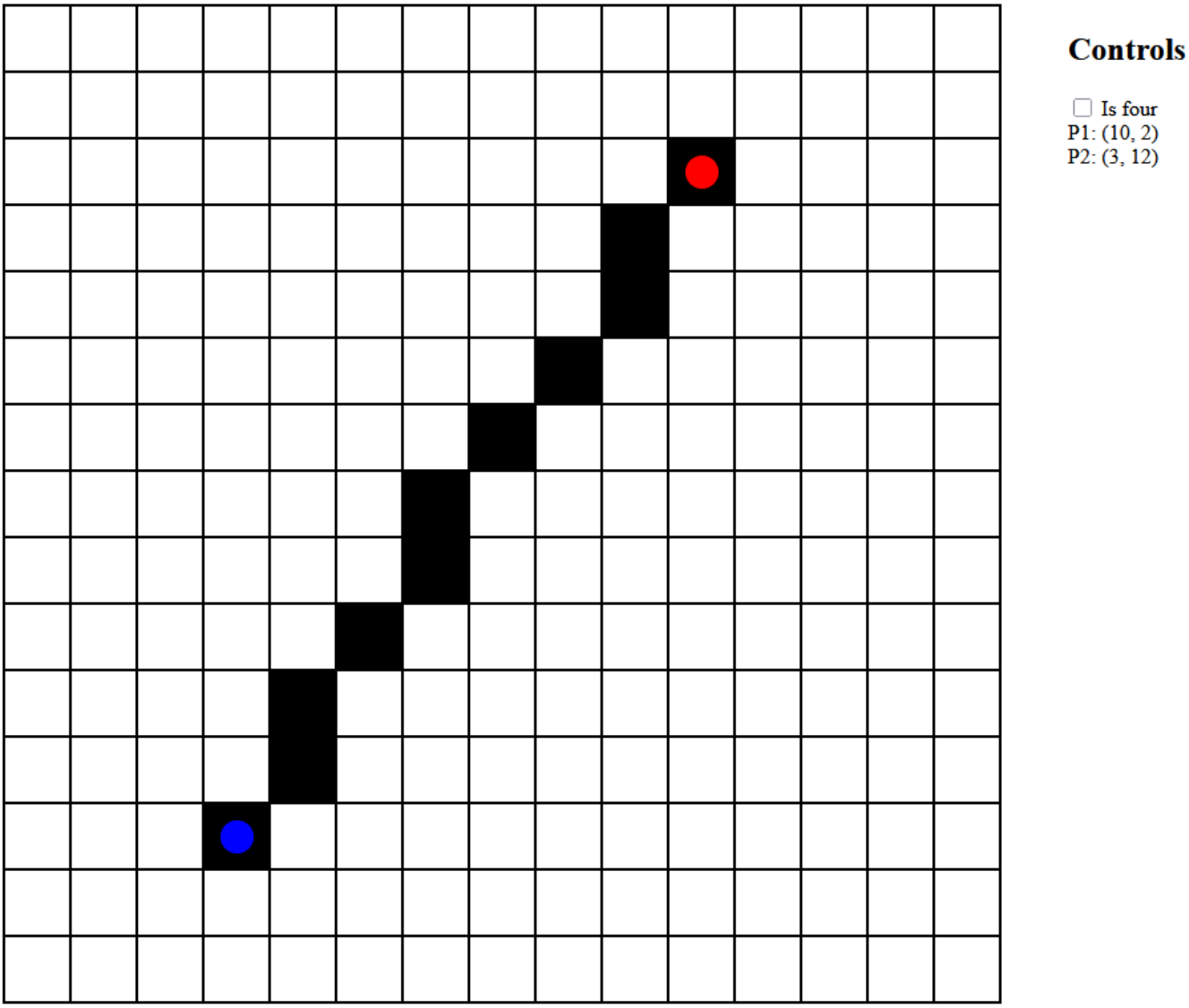

Рисунок 3 – Демонстрація роботи з іншими точками

3 Тест розробленої програми

```
const ROOT = document.querySelector("#root")
const p1Element = document.querySelector("#p1")
const p2Element = document.querySelector("#p2")

// Number of elements in row and col, all matrix is N*N
const N = 15

// Size of one cell, in px
const CELL = 50

document.documentElement.style.setProperty("--N", N);
document.documentElement.style.setProperty("--cell", `${CELL}px`);

let isFour = false
```

```
// map[int - Y]map[int - X]bool
let activeCells = { }

const P1 = { x: 2, y: 4 }
const P2 = { x: 11, y: 7 }
const points = { first: P1, second: P2 }

function isCellActive(x, y) {
	const yData = activeCells[y]
	if (!yData) return false

	const xData = yData[x]
	if (!xData) return false

	return true
}

function createPoint(x, y, order) {
	const point = document.createElement("div")
	point.classList.add("point")

	point.style.setProperty("--x", x);
	point.style.setProperty("--y", y);

	if (order == "first") {
		point.classList.add("point--fisrt")
	}

	if (order == "second") {
		point.classList.add("point--second")
	}

	return point
}

function render(points) {
	const grid = document.createElement("div")
	grid.className = "field"

	for (let row = 0; row < N; row++) {
		for (let col = 0; col < N; col++) {
			const cell = document.createElement("div")
			cell.className = "field-cell"

			if (isCellActive(col, row)) {
				cell.classList.add("field-cell--active")
			}

			cell.dataset.x = col
			cell.dataset.y = row

			grid.append(cell)
		}
	}

	grid.appendChild(createPoint(points.first.x, points.first.y, "first"))
	grid.appendChild(createPoint(points.second.x, points.second.y, "second"))
```

```
	p1Element.textContent = `P1: (${P1.x}, ${P1.y})`
	p2Element.textContent = `P2: (${P2.x}, ${P2.y})`
	ROOT.replaceChildren(grid)
}

function setPoint(x, y) {
	if (activeCells[y] == undefined) {
		activeCells[y] = {}
	}

	activeCells[y][x] = true
}

// Based on https://en.wikipedia.org/wiki/Bresenham%27s_line_algorithm
function line(x0, y0, x1, y1) {
	const dx = Math.abs(x1 - x0)
	const dy = -Math.abs(y1 - y0)

	const sx = x0 < x1 ? 1 : -1
	const sy = y0 < y1 ? 1 : -1

	let error = dx + dy

	while (true) {
		setPoint(x0, y0)
		const e2 = 2 * error

		if (x0 === x1 && y0 === y1) break

		if (!isFour) {
			if (e2 >= dy) {
				if (x0 == x1) break

				error += dy
				x0 += sx
			}

			if (e2 <= dx) {
				if (y0 == y1) break

				error += dx
				y0 += sy
			}
		}

		if (isFour) {
			if (e2 <= dx) {
				if (y0 == y1) break

				error += dx
				y0 += sy
			} else if (e2 >= dy) {
				if (x0 == x1) break

				error += dy
				x0 += sx
			}
		}
	}
```

```
}

document.querySelector("#four").addEventListener("click", e => {
	const target = e.target
	isFour = target.checked
	main()
})

document.addEventListener("click", e => {
	const target = e.target

	if (target.classList.contains("field-cell")) {
		P1.x = Number(target.dataset.x)
		P1.y = Number(target.dataset.y)

		main()
	}
})

document.addEventListener("contextmenu", e => {
	e.preventDefault()
	const target = e.target

	if (target.classList.contains("field-cell")) {
		P2.x = Number(target.dataset.x)
		P2.y = Number(target.dataset.y)
		main()
	}
})

function main() {
	console.log("Main!")

	activeCells = {}
	line(P1.x, P1.y, P2.x, P2.y)
	render(points)
}

main()


<!DOCTYPE html>
<html lang="en">
<head>
	<meta charset="UTF-8">
	<meta name="viewport" content="width=device-width, initial-scale=1.0">
	<title>Graphics task line</title>
	<link rel="stylesheet" href="style.css">
</head>
<body>
	<div class="layout">
		<div id="root">

		</div>

		<div>
			<h2>Controls</h2>
			<label>
			<input type="checkbox" id="four" />
```

```
                        Is four
                        </label>
                        <div id="p1" ></div>
                        <div id="p2" ></div>
                </div>
        </div>
        <script src="main.js"></script>
</body>
</html>


body {
        box-sizing: border-box;
}

.layout {
        display: flex;
        column-gap: 50px;
}

.field {
        position: relative;
        border: 1px solid black;
        display: grid;

        width: calc(var(--N) * var(--cell));
 height: calc(var(--N) * var(--cell));

        grid-template-columns: repeat(var(--N),var(--cell));
 grid-template-rows: repeat(var(--N),var(--cell));
}

.field-cell {
        border: 1px solid black;
}

.field-cell--active {
        background: black;
}

.point {
        position: absolute;

        width: calc(var(--cell)/2);
        height: calc(var(--cell)/2);

        border-radius: 100%;

        margin: calc(var(--cell)/4);
        top: calc(var(--y) * var(--cell));
        left: calc(var(--x) * var(--cell));
}

.point--fisrt {
        background: red;
}

.point--second {
        background: blue;}
```