УДК 519.6 **DOI: 10.20998/2222-0631.2025.01(8).08**

С. О. ДОВГИЙ, Г. Г. БУЛАНЧУК, О. М. БУЛАНЧУК

GPU-РЕАЛІЗАЦІЯ РОЗРАХУНКУ ПРАВОЇ ЧАСТИНИ СЛАР У МЕТОДІ ДИСКРЕТНИХ ВИХРОВИХ РАМОК ДЛЯ ЗАДАЧ ОБТІКАННЯ ТОНКИХ ПЛАСТИН

У роботі розглядається можливість використання графічного процесора (GPU) для розрахунку правої частини системи лінійних алгебраїчних рівнянь (СЛАР) у методі дискретних вихрових рамок при моделюванні обтікання тонких пластин ідеальною рідиною. При цьому швидкість руху вузлів вихрової пелени також розраховувалась на графічному процесорі. Для реалізації алгоритму розрахунку на GPU було використано мову програмування GLSL для обчислювальних шейдерів, що реалізується в межах стандарту OpenGL, починаючи із версії 4.3. Для обчислень на центральному процесорі (CPU) використовувалась мова програмування C# та фреймворк OpenTK. При реалізації алгоритмів на CPU розпаралелювання обчислень здійснювалося з використанням статичного методу Parallel.For. CPU-реалізація використовує числа подвійної точності, тоді як GPU-реалізація – числа одинарної точності і виключає умовні оператори для підвищення продуктивності. Проведено порівняльний аналіз точності та швидкості розрахунку задач обтікання пластинок різного розміру під різними кутами атаки. Результати чисельних експериментів показали, що при зниженні точності менш ніж на 1 % (за критеріями розподілу тиску та повної сили) вдається досягти значного прискорення розрахунків – до 75 разів всієї задачі залежно від кількості вихрових елементів. Істотне збільшення похибки спостерігається при збільшені розрахункового часу до $t_{max} = 6$, але на той момент застосування методу може бути некоректним, оскільки відбувається самоперетин вихрової пелени. При цьому саме перенесення обчислень правої частини СЛАР на GPU прискорює розрахунок всієї задачі приблизно в десять разів. Отримані результати підтверджують доцільність та ефективність перенесення розрахунків правої частини СЛАР на графічний процесор у задачах моделювання просторового обтікання пластин з використанням методу дискретних вихрових рамок.

Ключові слова: ідеальна рідина, обтікання пластин, метод дискретних вихрових рамок, C#, GLSL, OpenGL, OpenTK, GPU, формула Біо – Савара, ядро Ренкіна.

S. O. DOVGYI, G. G. BULANCHUK, O. M. BULANCHUK

GPU IMPLEMENTATION OF THE RIGHT-HAND SIDE COMPUTATION IN THE DISCRETE VORTEX METHOD FOR THIN PLATE FLOW SIMULATION

This work explores the use of a graphics processing unit (GPU) to compute the right-hand side of a system of linear algebraic equations (SLAE) within the discrete vortex method when modeling the flow around thin flat plates in an ideal fluid. In addition, the velocity of the vortex sheet nodes was also computed on the GPU. The algorithm for GPU-based computation was implemented using the GLSL programming language for compute shaders, supported in the OpenGL standard starting from version 4.3. For CPU-based calculations, the C# programming language and the OpenTK framework were used. Parallelization on the CPU was achieved using the static method Parallel.For. The CPU implementation operates with double-precision numbers, while the GPU implementation uses single-precision arithmetic and avoids conditional operators to improve performance. A comparative analysis of the accuracy and performance was carried out for flow simulations around plates of different sizes and angles of attack. The results of numerical experiments demonstrate that, with a loss in accuracy of less than 1 % (in terms of pressure distribution and total force), a significant speedup of up to 75 times can be achieved, depending on the number of vortex elements. A noticeable increase in error is observed as the simulation time increases $t_{max} = 6$; however, at that point, the method may become physically invalid due to self-intersection of the vortex sheet. It was found that transferring only the right-hand side SLAE computation to the GPU yields a 10× speedup of the overall calculation. The obtained results confirm the feasibility and effectiveness of offloading the right-hand side computation to the GPU for simulating three-dimensional plate flows using the discrete vortex method.

Key words: ideal fluid, flat plate flow, discrete vortex method, C#, GLSL, OpenGL, OpenTK, GPU, Biot–Savart formula, Rankine core-OpenGL, vortex segments, GPU, computer shaders, ideal fluid, vortex sheet.

Вступ. У роботі [1] було досліджено можливості *моделювання тривимірного обтікання* тонких пластин *методом дискретних вихрових рамок* [2] із використанням *графічного процесору GPU*. Було проведено порівняння координат пелени, обчислених за класичною схемою на *центральному процесорі CPU*, та розрахунками координат із обчисленням швидкості на GPU. Було досліджено відносну похибку при визначенні координат вузлів пелени для квадратної пластинки, яка розміщена перпендикулярно до набігаючого потоку. При цьому *ядро Ренкіна* у *формулі Біо – Савара* не використовувалося. У роботі [3] було досліджено точність розрахунку розподілу тиску на видовженій пластині, де рух пелени обчислювався на GPU. При цьому використовувався той самий алгоритм, але було застосовано ядро Ренкіна. Дослідження показали, що втрата точності порівняно з центральним процесором є досить малою.

Описаний у роботі [1] алгоритм моделювання включав наступні етапи:

1. Задання початкових умов та параметрів дискретизації.
2. Формування структур даних для збереження вузлових і контрольних точок на пластині, точок пелени.
3. Формування матриці методу дискретних вихрових рамок.
4. Розрахунок правої частини *системи лінійних алгебраїчних рівнянь (СЛАР)*.
5. Розв’язання СЛАР і визначення густини подвійного шару на поверхні пластини.
6. Формування вихрової пелени в початковий момент часу та задання густини подвійного шару на наступних кроках.
7. Обчислення швидкості в контрольних точках пелени.
8. Переміщення вихрових пелен за локальними швидкостями.

9. Повернення до кроку 4 в наступний момент часу.

У попередній реалізації було перенесено на GPU лише етап обчислення швидкостей у вузлах вихрових пелен (крок 7) з використанням обчислювальних шейдерів на мові GLSL. Усі обчислення на GPU виконувались з одинарною точністю (float), тоді як розрахунок правої частини СЛАР (крок 4) здійснювався на CPU на мові C# з використанням змінних подвійної точності (double). Результати порівняльних розрахунків показали, що при відносно малій кількості часових кроків відхилення у координатах вихрової пелени є незначними і візуально малопомітними.

У багатьох *практичних задачах*, зокрема при моделюванні *вітрової ситуації* в зонах міських забудов, кількість геометричних об'єктів (будинків) може бути досить великою. Хоча при тривалому моделюванні та зростанні кількості вихрових відрізків похибки поступово накопичуються, оптимізація обчислювальних витрат все ж залишається актуальною.

У зв'язку з цим постає питання про доцільність перенесення обчислення правої частини СЛАР (крок 4) також на графічний процесор (GPU). Однак такий перехід потребує детального аналізу впливу одинарної точності та GPU-специфічних обмежень (зокрема, уникнення умовних операторів) на точність і фізичну коректність обчислень гідродинамічних характеристик, таких як поле швидкостей та тиску.

Метою даної роботи була розробка та тестування GPU-реалізації обчислення правої частини СЛАР у методі дискретних вихрових рамок та визначення її впливу на точність та ефективність моделювання.

Розглядалося обтікання пластинок різного розміру під різними кутами атаки. Аналізувалися траєкторії вузлів вихрової пелени, розподіл тиску на поверхні пластинки та відповідні аеродинамічні сили. Для кожного випадку виконувалися два розрахунки за однакових початкових умов: перший – на центральному процесорі (CPU) з використанням подвійної точності, другий – із перенесенням розрахунку швидкостей у правій частині системи лінійних алгебраїчних рівнянь на графічний процесор (GPU). Окрім точності, оцінювалася також швидкість обчислень.

Постановка задачі. Розглядається *тривимірна задача* обтікання тонкої пластинки потоком *ідеальної нестисливої рідини*. Розв'язок задачі знаходиться у вигляді суми потенціалу зовнішнього потоку і потенціалу збурених пластинкою швидкостей. Потенціал збурених швидкостей повинен задовольняти *рівняння Лапласа* всюди, окрім поверхні пластинки та вихрових пелен, які сходять з пластинки [2]. При цьому на поверхні пластин в усі моменти часу повинна виконуватись *умова непротікання*.

Після *дискретизації* (розбиття пластинки на рамки) умова непротікання зводиться до системи лінійних алгебраїчних рівнянь відносно циркуляцій рамок. У кожен момент часу рамки сходять у потік, формуючи вихрові пелени, які переміщуються за місцевою швидкістю рідини. Швидкість в будь-якій точці обчислюється за формулою Біо – Савара.

Особливості реалізації формули Біо – Савара на GPU. Для точок, розташованих на великій відстані від осі вихрового відрізка, швидкість обчислювалась за формулою:

$$\vec{v} = \frac{\Gamma}{4\pi} \frac{\left[\Delta\vec{l} \times \vec{a}\right]}{\left\|\left[\Delta\vec{l} \times \vec{a}\right]\right\|^2} \Delta\vec{l} \cdot \left(\frac{\vec{a}}{a} - \frac{\vec{b}}{b}\right), \tag{1}$$

де $\Delta\vec{l}$ – вектор, напрямлений від початку відрізка до його кінця (рис. 1); $\vec{a} = \vec{r} - \vec{r}_0$ – вектор з початку вихрового відрізка $\vec{r}_0$ до точки спостереження $\vec{r}$, де визначається швидкість; $\vec{b} = \vec{a} - \Delta\vec{l}$ – вектор, проведений з кінця вихрового відрізка до точки спостереження.

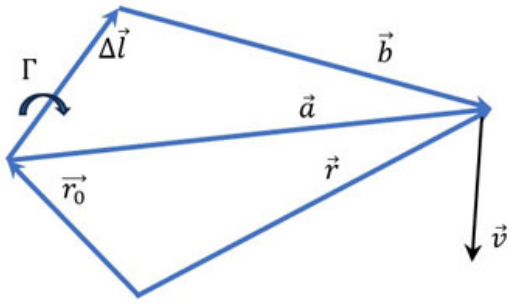


Рис. 1 – Швидкість від вихрового відрізка з використанням формули Біо – Савара.

Формула (1) дає значення швидкості, що прямує до нескінченності у випадку, коли точка спостереження

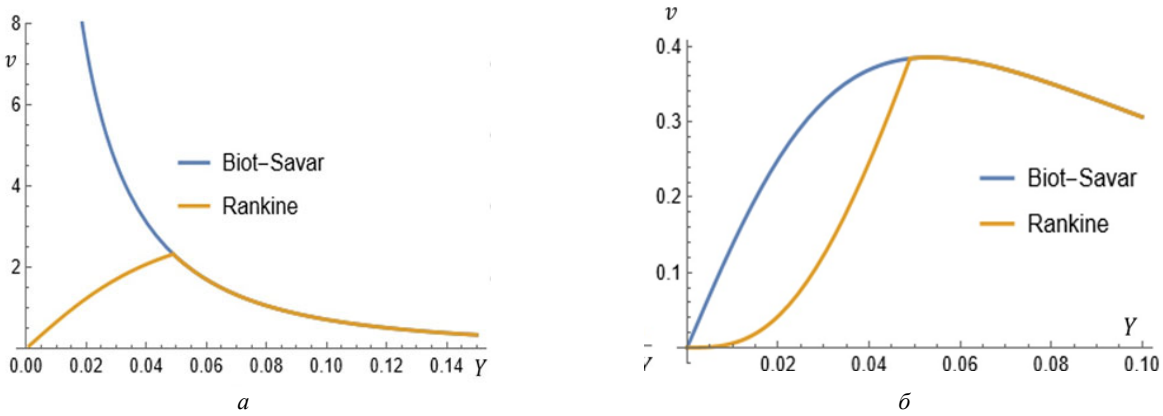
прямує до осі вихрового відрізка a її проекція потрапляє всередину цього відрізка (включаючи його кінці). Така особливість призводить до нефізичних результатів розрахунку течії рідини через *сингулярність* швидкості. Тому формулу (1) застосовують тоді, коли відстань до прямої, що проходить через відрізок, перевищує певне порогове значення R (радіус ядра Ренкіна):

$$\frac{\left\|\left[\Delta\vec{l}\times\vec{a}\right]\right\|^2}{\Delta l^2} > R^2. \tag{2}$$

Якщо ж точка спостереження розташована ближче до осі відрізка (умова (2) не виконується), застосовується регуляризована формула:

$$\vec{v} = \frac{\Gamma}{4\pi}\frac{\left[\Delta\vec{l}\times\vec{a}\right]}{\Delta l^2 R^2}\Delta\vec{l}\cdot\left(\frac{\vec{a}}{a}-\frac{\vec{b}}{b}\right). \tag{3}$$

Формула (3) дозволяє уникнути сингулярностей (рис. 2, *а*).



а *б*

Рис. 2 – Залежність швидкості $\boldsymbol{v}$ вихрового відрізка довжини $\Delta\boldsymbol{l} = 0.1$ від відстані до відрізка y.
Радіус Ренкіна $R = 0.49\Delta l$. *а* – вздовж перпендикуляра, що проходить через центр відрізка;
б – вздовж перпендикуляра, що проходить на відстані $\Delta\boldsymbol{l}$ від центра відрізка.

Слід зазначити, що існують також інші підходи для регуляризації розрахунків з використанням формули Біо – Савара, короткий огляд яких дано в роботі [4]. Варто звернути увагу, що для методу дискретних вихрових рамок регуляризація (3) дає занижені значення швидкостей на осі вихрового відрізка, що позитивно впливає на стабільність методу (рис. 2, *b*).

Нижче наведено фрагмент коду по реалізації формул (1) – (3) відповідно до геометричної конфігурації, представленої на рис. 1, на мові програмування C# з використанням подвійної точності.

```
    public static Vector3D Velocity(Vector3D r, Vector3D start, Vector3D dl, double gamma, double
rankine)
    {
1       double koef = 1.0 / (4 * Math.PI);
2       Vector3D a = r - start;
3       Vector3D b = a - dl;
4       Vector3D cross = dl.Cross(a);
5       double d = cross * cross;
6       if (d == 0.0)
7          return new Vector3D();
8       double ro = rankine * rankine;
9       double s = dl * (a / a.Abs() - b / b.Abs());
10      double bl = dl * dl;
11      if (d / bl > ro)
12          return gamma * cross * koef / d * s;
13      else
14          return gamma * cross * koef / (bl * ro) * s;
     }
```

Фрагмент коду 1. Розрахунок швидкості від вихрового відрізка за формулами (1) – (3) з урахуванням моделі ядра Ренкіна (мова C#).

Функція Velocity обчислює швидкість потоку в точці спостереження r ($\vec{r}$), що індукується вихровим відрізком з початком у точці start ($\vec{r}_0$) та вектором довжини dl ($\Delta\vec{l}$) з урахуванням згладжування за допомогою ядра Ренкіна, де gamma – циркуляція, що визначає інтенсивність вихрового відрізка, rankine – радіус ядра Ренкіна,

який використовується для згладжування швидкостей поблизу вихрового відрізка.

У рядку 6 перевіряється, чи знаходиться точка спостереження r на осі вихрового відрізка. У такому випадку векторний добуток $\left[\Delta\vec{l}\times\vec{a}\right]$ дорівнює нулю і повертається нульова швидкість (рядок 7). Це також запобігає діленню на нуль у рядку 9, оскільки за умови $\vec{a}=0$ або $\vec{b}=0$ код далі не виконується.

У рядку 11 здійснюється перевірка, чи знаходиться точка спостереження r на відстані, більшій ніж радіус Ренкіна, і якщо це так, то розрахунок проводиться за формулою Біо – Савара (1) у рядку 12. Якщо ж точка потрапляє у внутрішню область, де швидкість згладжується ядром Ренкіна, тоді застосовується формула (2), представлена у рядку 14.

Слід зазначити, що для обчислень на графічному процесорі аналогічна реалізація має бути адаптована до мови GLSL. У цьому випадку необхідно використовувати типи даних float та уникати умовних операторів if, оскільки їх використання суттєво знижує продуктивність на GPU, що підтверджується як експериментальними результатами, так і документацією NVIDIA.

Нижче наведено фрагмент коду по реалізації формул (1) та (2), відповідно до геометричної конфігурації, представленої на рис. 1, на мові програмування GLSL з використанням чисел одинарної точності.

```
#define PI 3.14159265
const float koef = 1.0/4.0/PI;
const float eps = 1.0e-14;

vec3 Velocity(vec3 r, vec3 start, vec3 dl, float gamma, float rankine)
{
1     vec3 a = r - start;        // Вектор від початку відрізка до точки спостереження
2     vec3 b = a - dl;           // Вектор до кінця відрізка
3     vec3 cross_dl_a = cross(dl, a); // Векторний добуток [Δl × a]
4     float d = dot(cross_dl_a, cross_dl_a);   // ||[Δl × a]||^2
5     float ro = rankine * rankine;            // R^2
6     float cross_selector = step(d, eps);     //Захист від ділення на нуль при малих d
7     float bl = dot(dl, dl); // Δl^2
8     float selector = step(ro, d / bl);       // Перемикач: чи використовувати ядро Ренкіна
9     float dist = selector * d + (1.0 - selector) * bl * ro; // Згладжена відстань
10    float la = length(a) + cross_selector;   // Довжина вектора a (із захистом)
11    float lb = length(b) + cross_selector;   // Довжина вектора b (із захистом)
12    float s = dot(dl, a)/la - dot(dl, b)/lb; // Скалярна частина формули Біо-Савара
13    return (1.0 - cross_selector) * gamma * koef * cross_dl_a * s / dist;
}
```

Фрагмент коду 2. Розрахунок швидкості від вихрового відрізка по формулі (1) – (3) з урахуванням моделі ядра Ренкіна (мова GLSL).

Для обчислення векторного добутку використовується вбудована функція **cross** (рядок 3), для скалярного добутку функція **dot** (рядок 4). У рядку 6 вводиться змінна **cross_selector** для заміни **if** у рядку 6 для коду на C#. Функція **step(d, eps)** поверне нуль, якщо **d** більше **eps** і одиницю, якщо менше. Таким чином, **cross_selector** дорівнює одиниці, якщо векторний добуток майже нуль і тоді рядок 13 поверне нуль. У рядках 10 та 11 **cross_selector** додається, щоб уникнути можливого ділення на нуль у рядку 12 (випадок коли точка спостереження лежить на кінці відрізка).

У рядку 8 обчислюється змінна, яка дозволяє уникнути використання оператора **if** у рядку 11 коду на C#. Змінна **selector** дорівнює одиниці, якщо відстань від осі відрізка більша за радіус Ренкіна. Тоді у рядку 9 перший доданок дорівнює **d**, а другий – нулю. Інакше перший доданок буде нуль, а другий **bl*ro**. Тоді рядок 13 буде повертати швидкість із врахуванням радіуса Ренкіна.

Розрахунок обтікання пластинки при малих кутах атаки та відривах пелени із трьох сторін. Було розглянуто обтікання пластини при малих кутах атаки α. Відрив відбувався з трьох сторін. Обчислення проводилися до моменту часу $t_{max}=4$, крок по часу $\Delta t=0.1$, просторова дискретизація $\Delta s=0.1$ (розмір сторони приєднаної рамки). Пластина має розмір 2×4 (перше значення – це ребро вздовж осі X, друге – по напрямку вітру, що напрямлений протилежно осі Z). Відносна похибка при розрахунку координат пелени розраховувалась за формулою:

$$\varepsilon_r=\frac{\left|\vec{r}_{GPU}-\vec{r}_{CPU}\right|}{\left|\vec{r}_{CPU}\right|}\cdot100\%$$

Відносна похибка при розподілі тиску по поверхні обчислювалась за формулою:

$$\varepsilon_p = \frac{|p_{GPU} - p_{CPU}|}{|p_{CPU}|} \cdot 100\%$$

Відносна похибка при обчисленні нормальної сили, що діє на пластинку – за формулою:

$$\varepsilon_F = \frac{|F_{GPU} - F_{CPU}|}{|F_{CPU}|} \cdot 100\%$$

У табл. 1 наведені результати розрахунку відносних похибок порівняно з центральним процесором. На рис. 3 представлена форма вихрової пелени за пластиною.

Таблиця 1 – Максимальні відносні похибки обчислень пластинки з трьома відривами від кута атаки

Кут атаки, градуси	Максимальна відносна похибка координат пелени: ε_r, %	Максимальна відносна похибка розподілу тиску: ε_p, %	Відносна похибка нормальної сили: ε_F, %
10	0.0003	0.00004	0.00004
20	0.008	0.017	0.0001
30	0.6	0.059	0.0041
40	4.1	1	0.12

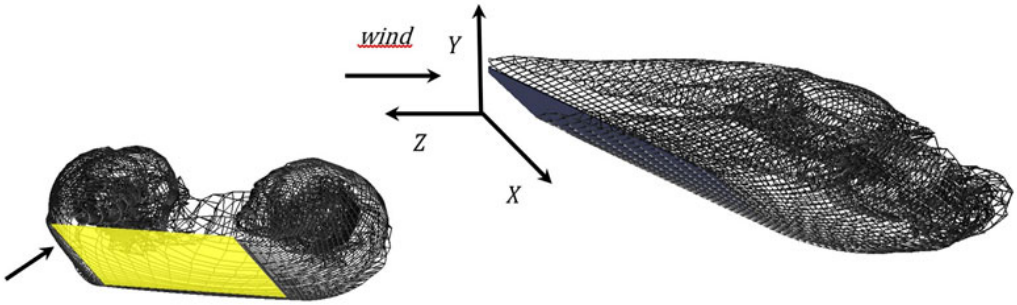


Рис. 3 – Форма пелени за пластинкою розміром 2×4 при куті атаки 40 градусів у момент часу $t_{max} = 4$ при розрахунку на графічному процесорі.

Із табл. 1 видно, що зі збільшенням кута атаки похибки обчислень усіх величин монотонно зростають. По координаті відносна похибка досягає 4.1 %, по розподілу тиску – 1 %, для нормальної сили – 0.12 %. Слід зазначити, що коли права частина не розраховувалась на GPU, а швидкості вузлів пелени при цьому рахувались на GPU, то відносна похибка по координатам пелени становила 2.8 %, по розподілу тиску – 0.73 %, для нормальної сили – 0.083 %.

Перевірявся також результат розрахунків до $t_{max} = 6$, коли кут атаки дорівнював 40°. У цьому випадку максимальна відносна похибка в положенні вузлів пелени досягала $\varepsilon_r = 19\%$, але максимальна похибка у розподілі тиску була досить незначною: $\varepsilon_p = 0.25\%$, а похибка у визначенні нормальної сили дорівнювала 0.0035 %.

Слід зазначити, що при розрахунку на такий великий проміжок часу застосування методу дискретних вихрових рамок може приводити до некоректних результатів, бо пелена руйнується і вихрові відрізки витягуються, що призводить до виникнення нефізичних швидкостей. У даному випадку безрозмірна швидкість перевищувала 3. Для уникнення такого ефекту у свій час було запропоновано і реалізовано *метод вставки проміжних точок* [5]. Але цей підхід призводив до значного збільшення кількості вихрових відрізків та часу розрахунку. В перспективі планується вирішити цю проблему із застосуванням графічних процесорів.

Розрахунок обтікання пластинки при великих кутах атаки та відривах пелени із чотирьох сторін. Розглядалась пластинка при великих кутах атаки (рис. 4). Відрив відбувався із чотирьох сторін. Обчислення проводилися до моменту часу $t_{max} = 1.6$, крок по часу дорівнював $\Delta t = 0.1$, крок просторової дискретизації – $\Delta s = 0.1$. Пластина мала розмір 8×2. У цьому випадку має місце значне наближення вихрової пелени до поверхні пластинки і зі збільшенням часу відбувається перетин пеленою поверхні пластинки, якщо не вживати спеціальних заходів.

На рис. 4 представлена форма пелени за пластинкою при кутах атаки $75^\circ, 80^\circ, 85^\circ, 90^\circ$ для моменту часу $t_{max} = 1.6$ при розрахунку на графічному процесорі.

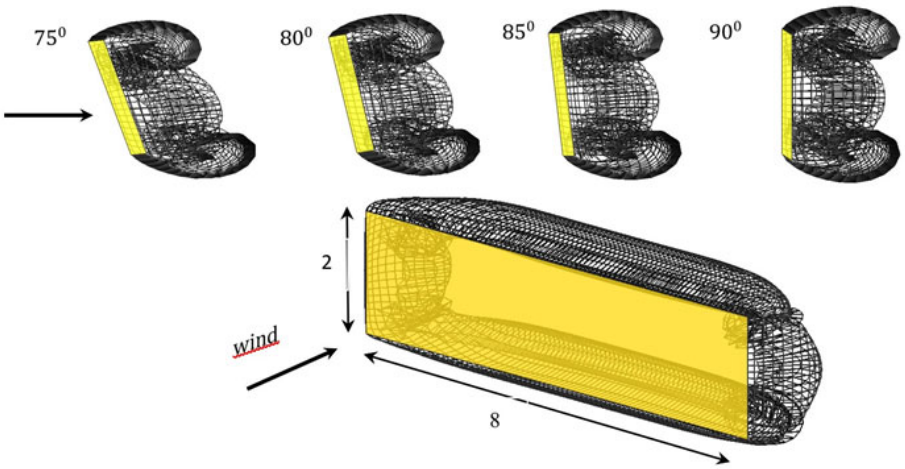


Рис. 4 – Форма пелени за пластинкою розміром 8×2 при кутах атаки $75^{\circ}, 80^{\circ}, 85^{\circ}, 90^{\circ}$ для моменту часу $t_{max}=1.6$ при розрахунку на графічному процесорі.

Таблиця 2 – Залежність похибок обчислень для пластинки з чотирма відривами при різних кутах атаки

Кут атаки, градуси	Максимальна відносна похибка координат пелени: ε_r, %	Максимальна відносна похибка розподілу тиску: ε_p, %	Відносна похибка нормальної сили: ε_F, %
75	0.051	0.068	0.00024
80	0.043	0.029	0.00028
85	0.054	0.023	0.00033
90	0.063	0.010	0.00036

У табл. 2 представлені похибки обчислень із використанням GPU порівняно з центральним процесором при різних кутах атаки. Із табл. 2 видно, що максимальне значення відносної похибки координат вузлів пелени досягає 0.063 % при куті атаки 90°, максимальна відносна похибка тиску – 0.068 % при $\alpha=75^{\circ}$, максимальна відносна похибка нормальної сили – 0.00036 % при $\alpha=90^{\circ}$. Похибки настільки малі, що фактично їх можна не брати до уваги.

Із рис. 4 видно, що для всіх розглянутих кутів атаки пелени залишаються досить гладкими.

Швидкість розрахунків для пластинок різних розмірів. Порівняння швидкості обчислень здійснювалося із використанням графічного процесора nVIDIA GeForce RTX 2060 (архітектура Turing, 6 ГБ відеопам'яті, 30 потокових мультипроцесорів, 1920 ядер CUDA, тактова частота 1680 МГц) та центрального процесора AMD Ryzen 5 3600 (тактова частота 3925 МГц, 6 фізичних ядер, 12 логічних потоків). Об'єм оперативної пам'яті системи становив 32 ГБ.

Зазначимо, що обчислення правої частини системи лінійних алгебраїчних рівнянь (СЛАР) та швидкостей у вузлах пелени також були розпаралелені на центральному процесорі за допомогою конструкції Parallel.For. Відповідно до даних, отриманих із диспетчера завдань (Resource Manager), при цьому використовувалися всі 12 логічних потоків процесора.

Таблиця 3 – Швидкість обчислень пластинок різних розмірів

Розмір пластинки	Час обчислення правої частини, ms (CPU/GPU)	Прискорення (для правої частини СЛАР)	Загальний час обчислень, ms (CPU/GPU)	Загальне прискорення	Загальне прискорення без правої частини на GPU
1×4	153/10	15	28179/403	70	7.6
2×4	337/19	18	44230/818	54	5.7
2×6	689/25	28	89512/1359	66	5.8
2×8	1105/33	33	151321/2013	75	5.7

Обчислення виконувалися до моменту часу $t_{max}=4$ із кроком по часу $\Delta t=0.1$ та просторовою дискретизацією $\Delta s=0.1$. Кут атаки становив 40°, моделювалися три відриви. Час, необхідний для обчислення правої частини СЛАР, наведено для останнього кроку по часу.

У табл. 3 представлена швидкість обчислень для пластинок різних розмірів, на рис. 5 – форма вихрової пелени.

Із табл. 3 видно, що прискорення обчислень правої частини СЛАР на графічному процесорі порівняно із центральним процесором зростає від 15 до 33 разів зі збільшенням розміру пластинки (кількості приєднаних рамок). Це свідчить про те, що при невеликій кількості рамок накладні витрати на передачу та копіювання даних між CPU та GPU становлять значну частину загального часу. Загальне прискорення моделювання коливається від 54 до 75 разів і досягає максимуму для найбільшої пластинки, де кількість елементів є максимальною. Завищене значення прискорення для найменшої пластинки пов'язане із особливостями розпаралелювання на CPU при розрахунку швидкостей: коли вихрових відрізків мало, а витрати на створення нових потоків такі самі. Починаючи із другого набору даних, спостерігається монотонне зростання прискорення. Також видно, що використання GPU для розрахунку правої частини СЛАР прискорює обчислення більш ніж у 10 разів при великих розмірах пластинки.

Із рис. 5 видно, що навіть при максимальній похибці розрахунків форма пелени залишається досить гладкою.

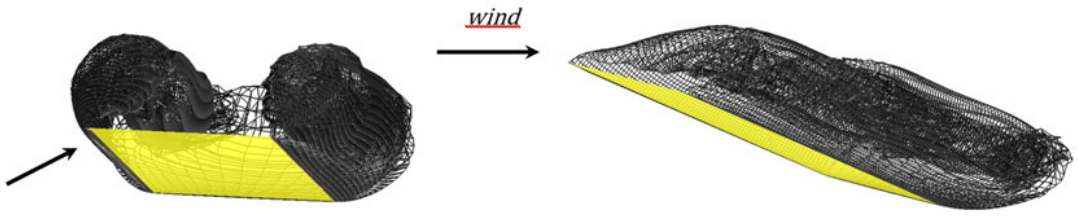


Рис. 5 – Форма пелени за пластинкою розміром 2×8 при куті атаки 40° у момент часу $t_{max}=4$, отримана у результаті розрахунків на графічному процесорі.

Перспективи подальших досліджень. Автори вважають, що дані дослідження можуть бути використані для вдосконалення методу дискретних вихрових рамок з використанням методики вставки проміжних точок. Ця методика дозволяє уникати надмірного подовження вихрових відрізків та зменшує пульсації швидкості та тиску, але за рахунок ускладнення алгоритму та значного збільшення часу розрахунку. Перенесення розрахунків швидкості та правої частини на GPU дозволить значно прискорити швидкість обчислень та одержувати покращені результати за розумний час.

Висновки. На основі проведених досліджень можна зробити такі висновки:

1. При малих кутах атаки похибка обчислень із використанням графічного процесора є незначною (менше 0.1 %) і монотонно зростає зі збільшенням часу розрахунку через накопичення похибок округлення. Найбільшу похибку можна спостерігати для координат вузлів пелени при великих кутах атаки. Водночас похибки при розрахунку тиску та нормальної сили залишаються меншими за 1 %. Це пояснюється тим, що пелена зноситься набігаючим потоком, і найбільш впливові рамки з великою циркуляцією розташовуються далеко від пластинки, що зменшує їхній вплив на тиск. Але навіть за умов найбільшої похибки форма пелени залишається візуально гладкою та фізично коректною.
2. При великих кутах атаки, але для відносно короткого часу моделювання (довший час спричиняє перетин пеленою пластинки), відносна похибка становить менше 0.1 %. Форма пелени залишається гладкою для всіх розглянутих кутів атаки.
3. Прискорення обчислень на графічному процесорі порівняно із центральним процесором досягає максимуму (75 разів) для найбільшої пластинки, де кількість елементів є максимальною. При максимальній похибці при розрахунку координат пелени її форма залишається гладкою.

Таким чином, перенесення обчислень правої частини системи лінійних алгебраїчних рівнянь на графічний процесор хоча й призводить до незначної втрати точності, проте забезпечує суттєве прискорення розрахунків, що робить запропонований підхід ефективним для практичного застосування при обчислення задач тривимірного обтікання великої кількості об'єктів методом дискретних вихрових рамок.

Список літератури

1. *Довгий С. О.*, *Буланчук Г. Г.*, *Буланчук О. М.*, *Листопадова В. В.* Точність розрахунків поля швидкостей від системи вихрових відрізків при використанні графічних процесорів // Вісник НТУ «ХПІ». Серія: Математичне моделювання в техніці та технологіях. – 2023. – №1. – С. 105 – 109. DOI: 10.20998/2222-0631.2023.01.15.

2. *Довги С. А.*, *Лифанов И. К.* Методы решения интегральных уравнений. – К. : Наукова думка, 2002. – 343 с.
3. *Довгий С. О.*, *Буланчук Г. Г.*, *Буланчук О. М.* Дослідження динамічних характеристик при моделюванні обтікання пластини з використанням графічних процесорів // IX Міжнар. конф. «Комп'ютерна гідромеханіка» 1 – 2 жовтня. – Київ : Інститут гідромеханіки НАН України, 2024. – С. 37 – 38.
4. *Abedi H.* Development of Vortex Filament Method for Wind Power Aerodynamics. PhD thesis, Div. of Fluid Dynamics. // Dept. of Applied Mechanics. – Chalmers University of Technology, Goteborg, Sweden, 2016. – 58 p.
5. *Буланчук Г. Г.*, *Буланчук О. Н.*, *Довгий С. А.* Метод дискретных вихревых рамок со вставкой промежуточных точек на вихревой пелене // Вісник Харківського національного університету. Серія : «Математичне моделювання. Інформаційні технології. Автоматизовані системи управління». – 2009. – Вип. 12. – № 863. – С. 36 – 46.

References (transliterated)

1. Dovgiy S. O., Bulanchuk G. G., Bulanchuk O. M., Lystopadova V. V. Tochnist' rozrakhunkiv polya shvydkostey vid systemy vykhrobykh vidrizkiv pry vtkorystanni grafichnykh protsesoriv [Accuracy of calculations of the velocity field from the system of vortex segments using graphic processors]. *Visnyk NTU "KhPI". Seriya : Matematychne modelyuvannya v tekhnitsi ta tekhnologiyakh* [Bulletin of the NTU "KhPI". Series: Mathematical modeling in engineering and technologies]. 2023, no. 1, pp. 105–109. DOI: 10.20998/2222-0631.2023.01.15.
2. Dovgiy S. A., Lyfanov I. K. *Metody resheniya integral'nykh uravneniy* [Methods of solving integral equations]. Kyiv, Naukova dumka Publ., 2002. 343 p.
3. Dovgiy S. O., Bulanchuk G. G., Bulanchuk O. M. Doslidzhennya dynamichnykh kharakterystyk pry modelyuvanni obtikannya plastyny z vykorystannyam grafichnykh protsesoriv [Study of Dynamic Characteristics in Plate Flow Simulation Using GPUs]. *IX Mignar. Konf. "Komp"yuterna gidromekhanika" 1 – 2 zhovtnya* [IX Intern. Conf. "Computer Hydromechanics", October 1 – 2]. Kyiv, Instytut gidromekhaniky NAN Ukrayiny Publ., 2024. pp. 37–38.
4. Abedi H. Development of Vortex Filament Method for Wind Power Aerodynamics. PhD thesis, Div. of Fluid Dynamics. *Dept. of Applied Mechanics*. Chalmers University of Technology, Goteborg, Sweden, 2016. 58 p.
5. Bulanchuk G. G., Bulanchuk O. M., Dovgiy S. O. Metod diskretnykh vikhrevykh ramok so vstavkoy promezhutochnykh tochek na vikhrevoy pelene [Discrete Vortex Frame Method with Intermediate Point Insertion on the Vortex Sheet]. Visnyk Kharkivs'kogo universytetu. Seriya : Matematychne modelyuvannya. Informatsiyni tekhnologiyi. Avtomatyzovani systemy upravlinnya [Bulletin of V.N. Karazin Kharkiv National University. Series : "Mathematical modeling. Information technology. Automated control systems"]. 2009, Vol. 12, no. 863, pp. 36–46.

Надійшла (received) 15.03.2025

Відомості про авторів / Information about authors

Довгий Станіслав Олексійович – доктор фізико-математичних наук, професор, академік НАН України, президент Малої академії наук України, м. Київ; тел.: (044) 489-55-99; ORCID: https://orcid.org/0000-0003-1078-0162; e-mail: s.dovgii@gmail.com.

Dovgyi Stanislav Oleksiiovych – Doctor of Physical and Mathematical Sciences, Professor, academician of the National Academy of Sciences of Ukraine, prezident of the Junior Academy of Sciences of Ukraine, Kyiv; tel.: (044) 489-55-99; ORCID: https://orcid.org/0000-0003-1078-0162; e-mail: s.dovgii@gmail.com.

Буланчук Галина Григорівна – кандидат фізико-математичних наук, доцент кафедри вищої та прикладної математики ДВНЗ «Приазовський державний технічний університет», м. Дніпро; тел.: (098) 201-83-08; ORCID: https://orcid.org/0000-0002-9871-2748; e-mail: ggbulan7@gmail.com.

Bulanchuk Galyna Hryhorivna – Candidate of Physical and Mathematical Sciences, Associate Professor, Pryazovskyi State Technical University, Dnipro; tel.: (098) 201-83-08; ORCID: https://orcid.org/0000-0002-9871-2748; e-mail: ggbulan7@gmail.com.

Буланчук Олег Миколайович – кандидат фізико-математичних наук, доцент, методист лабораторії математичних наук, Науково-методичний центр НЦ «Мала академія наук України», м. Київ; тел.: (096) 347-56-60; ORCID: https://orcid.org/0000-0002-2801-2244; e-mail: obulan65@gmail.com.

Bulanchuk Oleh Mykolaiovych – Candidate of Physical and Mathematical Sciences, Associate Professor, methodologist at the Laboratory of Mathematical Sciences, Scientific and Metodological Center of the National Center of the Small Academy of Sciences of Ukraine, Kyiv; tel.: (096) 347-56-60; ORCID: https://orcid.org/0000-0002-2801-2244; e-mail: obulan65@gmail.com.