УДК 004.032.26
DOI: 10.26102/2310-6018/2025.51.4.051

Оптимизация вычислений в модифицированном фильтре Калмана для обработки изображений с автокоррелированными помехами: сравнительный анализ методов

И.Н. Осипенко✉

Ульяновский государственный университет, Ульяновск, Российская Федерация

Резюме. В статье рассматривается задача оптимизации вычислительных затрат при использовании модифицированного фильтра Калмана для подавления автокоррелированного шума в цифровых изображениях. Наличие такого шума является характерной особенностью многих практических задач, включая медицинскую диагностику, дистанционное зондирование Земли и обработку видео в реальном времени. Классический дискретный фильтр Калмана, будучи оптимальным по критерию минимизации среднеквадратичной ошибки, недостаточно эффективен в условиях автокоррелированных помех. В этом случае решение задачи дискретной фильтрации может быть получено путем расширения вектора состояния за счет включения дополнительных переменных, описывающих структуру шумовой компоненты. Такой подход обеспечивает более точное восстановление исходного сигнала, но приводит к резкому росту вычислительной сложности, обусловленному увеличением размерности матриц и возрастанием требований к объему памяти. С целью снижения вычислительных затрат в статье проведен анализ трех различных стратегий оптимизации алгоритма фильтрации: использование разреженных матричных представлений, позволяющих существенно сократить количество операций при хранении и обработке данных; применение многопоточной обработки на CPU для повышения степени параллелизма вычислений; а также перенос вычислительно затратных процедур на графические процессоры (GPU). Экспериментальная часть работы включает тестирование разработанных алгоритмов на датасете CIFAR-10, к изображениям которого был добавлен искусственно сгенерированный автокоррелированный шум. Результаты показали, что наибольший прирост производительности достигается в GPU-реализации (ускорение в 6–7 раз по сравнению с базовой схемой), тогда как эффективность многопоточности и работы с разреженными структурами зависит от размера выборки и свойств данных. Полученные выводы подтверждают перспективность применения предложенных решений в высокопроизводительных системах обработки изображений и их интеграции в современные методы машинного обучения.

Ключевые слова: фильтр Калмана, модифицированный фильтр Калмана, автокоррелированный шум, обработка изображений, оптимизация вычислений, разреженные матрицы, параллельные вычисления, GPU-ускорение, CIFAR-10, машинное обучение.

Для цитирования: Осипенко И.Н. Оптимизация вычислений в модифицированном фильтре Калмана для обработки изображений с автокоррелированными помехами: сравнительный анализ методов. *Моделирование, оптимизация и информационные технологии.* 2025;13(4). URL: https://moitvivt.ru/ru/journal/pdf?id=2102 DOI: 10.26102/2310-6018/2025.51.4.051

Optimization of computations in the modified Kalman filter for image processing with autocorrelated noise: a comparative analysis of methods

I.N. Osipenko✉

Ulyanovsk State University, Ulyanovsk, the Russian Federation

Abstract. This paper addresses the problem of optimizing computational costs in the modified Kalman filter used for suppressing autocorrelated noise in digital images. Such noise is a common feature of many practical applications, including medical imaging, Earth remote sensing, and real-time video processing. The classical discrete Kalman filter, while being optimal in terms of minimizing the mean square error, is insufficiently effective under autocorrelated disturbances. In this case, a solution to the discrete filtering problem can be obtained by extending the state vector through the inclusion of additional variables that describe the structure of the noise component. This approach enables more accurate signal restoration but leads to a sharp increase in computational complexity due to the growth of matrix dimensionality and memory requirements. To reduce these computational costs, three optimization strategies for optimizing the filtering algorithm are analyzed: the use of sparse matrix representations, which significantly reduce the number of operations for data storage and processing; multithreaded processing on CPUs to increase computational parallelism; and the transfer of computationally intensive procedures to graphics processing units (GPUs). The experimental study involves testing the developed algorithms on the CIFAR-10 dataset, to which artificially generated autocorrelated noise was added. The results demonstrate that the greatest performance gain is achieved with the GPU-based implementation (a 6–7× speedup compared to the baseline scheme), while the effectiveness of multithreading and sparse matrices depends on the dataset size and structure. The findings confirm the potential of the proposed solutions for practical use in high-performance image filtering systems and their integration into modern machine learning methods.

Keywords: Kalman filter, modified Kalman filter, autocorrelated noise, image processing, computational optimization, sparse matrices, parallel computing, GPU acceleration, CIFAR-10, machine learning.

For citation: Osipenko I.N. Optimization of computations in the modified Kalman filter for image processing with autocorrelated noise: a comparative analysis of methods. *Modeling, Optimization and Information Technology*. 2025;13(4). (In Russ.). URL: https://moitvivt.ru/ru/journal/pdf?id=2102 DOI: 10.26102/2310-6018/2025.51.4.051

Введение

Современные задачи цифровой обработки изображений в реальном времени требуют использования высокоэффективных алгоритмов, способных обрабатывать большие объемы данных с высокой скоростью при ограниченных вычислительных ресурсах [1]. Фильтр Калмана, являющийся оптимальным алгоритмом по критерию минимизации среднеквадратичной ошибки, представляет особый интерес в контексте задач подавления шумов [2], однако при обработке больших массивов данных в реальном времени возрастающая вычислительная сложность существенно ограничивает его практическое применение.

В зарубежных исследованиях показана высокая эффективность фильтра Калмана при обработке изображений и видео. Так, Sun и соавторы [3] предложили методику обработки видео при двухфотонной флуоресцентной микроскопии, основанную на модифицированном фильтре Калмана, что позволило подавить фоновый шум и повысить качество восстановления изображений микрообъектов; уровень правильной детекции при этом достигал 93 % для бактерий и 98 % для вирусов. В работе Roy и Mitra [4] была предложена модель обнаружения визуальной салиентности в статичных изображениях, где фильтр Калмана использовался для выделения аномальных областей.

Среди отечественных исследований следует отметить работу Сироты и Иванкова [5], в которой рассмотрены блочные алгоритмы восстановления изображений с применением фильтра Калмана для задач сверхразрешения. Кроме того, Ионов, Болдырёхин и соавторы [6] описали использование фильтра Калмана для фильтрации полутоновых изображений в системах сельскохозяйственного мониторинга, разработав алгоритм и программное обеспечение для устранения шумов.

Ранее проведенные эксперименты [7] подтвердили эффективность классического дискретного фильтра Калмана при подавлении коррелированного шума, что позволило повысить качество входных данных перед обучением автоэнкодеров. В последующих исследованиях [8] предложена модификация метода, основанная на расширении вектора состояния для более точного учета автокоррелированных шумовых компонент. Показано, что включение дополнительных переменных повышает точность восстановления сигнала, однако приводит к существенному росту вычислительных затрат, что ограничивает практическое применение метода при обработке больших объемов данных.

Актуальность настоящего исследования определяется необходимостью разработки эффективных методов оптимизации вычислений в модифицированном фильтре Калмана [9], предназначенном для устранения автокоррелированного шума, которые позволят сохранить его статистические преимущества при значительном повышении производительности. В работе рассматриваются три принципиальных подхода к оптимизации:

- применение разреженных матричных представлений [2, 10];
- параллельная обработка на многоядерных CPU [11];
- перенос вычислений на графические процессоры (GPU) [12, 13].

Методология исследования включает теоретический анализ вычислительной сложности, реализацию оптимизированных алгоритмов, их экспериментальное сравнение на стандартном наборе данных CIFAR-10 с искусственно добавленным автокоррелированным шумом, а также оценку полученной производительности.

Полученные результаты могут быть использованы при разработке высокоэффективных алгоритмов фильтрации изображений [14] и их интеграции в системы машинного обучения, требующие устойчивости к шумам и высокой вычислительной эффективности.

Материалы и методы

В разделе изложены постановка задачи дискретной фильтрации изображений с автокоррелированными помехами, алгоритмическая структура программы, математическая модель измеряемых данных, описание модифицированного фильтра Калмана и методы оптимизации вычислений.

Общая структура алгоритма представлена на Рисунке 1. На вход подаются исходные изображения, к которым добавляется искусственно сгенерированный автокоррелированный шум. Далее последовательно выполняются этапы прогнозирования состояния, обновления ковариационной матрицы ошибок, вычисления коэффициента Калмана и коррекции оценки состояния. В отличие от классического фильтра Калмана, схема дополнена блоком расширения вектора состояния, позволяющим учитывать автокоррелированную природу шумовой компоненты.

Представленная на Рисунке 1 схема отражает ключевые этапы функционирования алгоритма и служит основой для дальнейшей формализации задачи дискретной фильтрации изображений с автокоррелированными помехами.

При решении задачи цифровой обработки изображений возникает необходимость учитывать наличие автокоррелированных шумов наряду с некоррелированными и/или коррелированными шумами. Цель дискретной фильтрации заключается в восстановлении истинного состояния изображения x_k по наблюдениям y_k, содержащим автокоррелированные помехи. Для этого классический фильтр Калмана требует модификации, так как его оптимальность обеспечивается только в случае белого шума измерений.

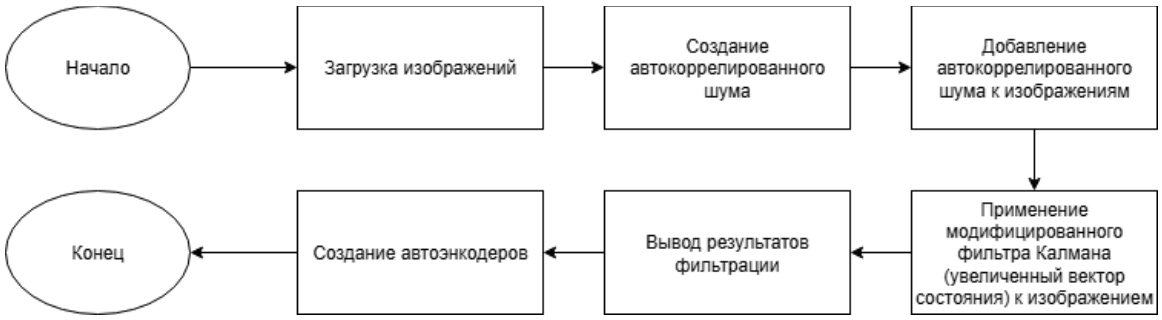

Рисунок 1 – Алгоритмическая схема работы программы, реализующей модифицированный фильтр Калмана с расширением вектора состояния
Figure 1 – Algorithmic scheme of the program that implements a modified Kalman filter with extended state vector

Математическая модель измеряемых данных. В общем виде динамика системы и процесс измерений описываются уравнениями:

$$x_{k+1} = Ax_k + \omega_k, \tag{1}$$

$$y_k = Cx_k + \upsilon_k, \tag{2}$$

где $x_k \in R^n$ – скрытый вектор состояния, подлежащий оцениванию; $\omega_k \in R^n$ – шум в уравнении объекта; $y_k \in R^p$ – вектор измерений; $\upsilon_k \in R^p$ – автокоррелированный шум. В отличие от классической постановки, шум υ_k не является белым, а имеет конечный шаг автокорреляции l и задается выражением:

$$\upsilon_k = \sum_{i=0}^{l} H_i \varsigma_{k-i}, k = 0,1,\ldots, \tag{3}$$

где $\varsigma_k \in R^q$ – независимые гауссовские случайные векторы, H_i – матрицы коэффициентов.

Алгоритм модифицированного фильтра Калмана. Для учета автокоррелированного шума вводится расширенный вектор состояния [9]:

$$\varphi_k = \begin{bmatrix} x_k \\ \varsigma_k \\ \varsigma_{k-1} \\ \vdots \\ \varsigma_{k-l+1} \end{bmatrix}. \tag{4}$$

Тогда динамика расширенной системы описывается уравнениями вида (5), (6) с матрицами (7), имеющими блочную структуру:

$$\varphi_{k+1} = A_o\varphi_k + \varepsilon_k, \tag{5}$$

$$y_k = C_o\varphi_k + \eta_k, \tag{6}$$

где

$$A_o = \begin{pmatrix} A & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & I & 0 & 0 & 0 & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & \ldots & 0 & 0 & I & 0 \end{pmatrix}, C_o = (C, H_o, H_1, \ldots, H_l). \tag{7}$$

Для такой расширенной модели динамической системы применяются стандартные рекуррентные шаги фильтра Калмана: прогноз, обновление ковариационной матрицы, вычисление коэффициента Калмана K_k и коррекция оценки. Полная формулировка алгоритма дискретной фильтрации дана в [9].

Особенностью данного подхода является рост размерности задачи при увеличении конечного шага автокоррелированного шума в измерениях. Как отмечено в работе [15], расширенный вектор состояния включает в себя как исходный вектор $x_k \in R^n$, так и последовательность шумовых векторов $\varsigma_k, \varsigma_{k-1}, \dots, \varsigma_{k-l+1}$, каждый из которых имеет размерность q. Таким образом, итоговая размерность расширенного состояния равна:

$$\tilde{n} = n + (l+1)q, \tag{8}$$

что непосредственно следует из построения вектора φ_k. При увеличении параметра l, характеризующего шаг автокорреляции, размерность системы возрастает линейно, а вычислительная сложность операций фильтрации – кубически по $\tilde{n}$, что приводит к значительному росту времени вычислений и объема используемой памяти, что делает оптимизацию вычислений особенно актуальной.

Перед детальным анализом методов оптимизации важно учитывать, что выбор стратегии ускорения вычислений в модифицированном алгоритме Калмана зависит от трех ключевых аспектов: структуры обрабатываемых данных, характеристик вычислительного оборудования и требований к качеству обработки. Рассмотрим основные подходы, наиболее распространенные в современных исследованиях.

Одним из перспективных подходов к оптимизации вычислений в модифицированном алгоритме Калмана является применение разреженных матриц [2]. Теоретически, матрицы, участвующие в вычислениях (например, матрица перехода состояния, ковариационные матрицы шума и ошибок), могут обладать значительной долей нулевых элементов, что позволяет сократить объем хранимых данных и ускорить выполнение операций линейной алгебры [10].

Эффективность метода зависит от степени разреженности матриц. В алгоритмах калмановской фильтрации матрицы перехода состояния и ковариационные матрицы могут быть плотными, поэтому перед применением необходим предварительный анализ структуры матриц, включая статистическую оценку доли нулевых элементов, визуализацию структуры и спектральный анализ для оценки блоков коррелированности [16].

Для ускорения обработки изображений в алгоритме Калмана можно задействовать многопоточные вычисления на центральных процессорах [12]. Современные CPU оснащены несколькими физическими и логическими ядрами, что позволяет распределять вычисления между потоками и выполнять операции параллельно.

В реализации, использованной в данной работе, многопоточная обработка построена на основе библиотеки joblib, где параметр $\mathrm{n\_jobs} = -1$ задает автоматическое использование всех доступных логических ядер. Такой механизм исключает необходимость ручного выбора числа потоков и адаптирует выполнение под архитектуру конкретного процессора. При этом внутренние механизмы joblib ограничивают избыточное распараллеливание, учитывая накладные расходы на распределение задач.

Преимущества многопоточного выполнения:

– использование стандартных инструментов Python, таких как joblib и multiprocessing, для удобной организации параллельных вычислений;

– эффективная загрузка всех ядер процессора, особенно при работе с большими объемами данных;

– улучшение масштабируемости алгоритма при увеличении количества доступных вычислительных ресурсов.

В то же время распараллеливание не всегда обеспечивает линейный прирост производительности: создание потоков, обмен данными и доступ к общей памяти создают дополнительные накладные расходы. Если узким местом является пропускная

способность памяти, увеличение числа потоков может не привести к заметному ускорению [1].

Перенос вычислений на графические процессоры является наиболее эффективным способом оптимизации в задачах, требующих массового параллелизма. GPU содержит тысячи потоковых процессоров, что позволяет выполнять вычисления значительно быстрее по сравнению с CPU, особенно при работе с большими матрицами [17, 11].

GPU-ускорение реализуется через библиотеку CuPy [17], которая повторяет интерфейс NumPy, но выполняет вычисления на GPU. Основные преимущества метода:

– высокая степень параллелизма, позволяющая ускорить операции с большими матрицами [12, 17];

– эффективное выполнение линейной алгебры, включая матричные умножения и обращение матриц [17, 11];

– снижение нагрузки на CPU благодаря переносу вычислений на GPU [17, 13].

Применение GPU требует наличия соответствующего аппаратного обеспечения и дополнительных затрат на управление памятью, поскольку передача данных между CPU и GPU может стать узким местом из-за латентности шины PCIe при частой передаче больших массивов.

Для объективной оценки эффективности рассмотренных методов оптимизации разработан комплексный экспериментальный протокол, включающий несколько взаимосвязанных этапов. Важно подчеркнуть, что достоверность результатов сравнения в значительной степени зависит от корректности выбора тестовых условий и методик измерений.

Основной целью эксперимента является измерение времени выполнения алгоритма при использовании стандартного CPU, многопоточной обработки, разреженных матриц и ускоренных вычислений на GPU.

Эксперименты проводились на вычислительной платформе со следующими характеристиками:

– процессор (CPU): AMD Ryzen 9 5900X 12-Core Processor;

– оперативная память (RAM): 48GB DDR4;

– графический процессор (GPU): NVIDIA GeForce RTX 4070 SUPER (ядра CUDA 7168);

– программное обеспечение: Python 3.11, CUDA 11.8, cuDNN 90100;

– среда выполнения: PyCharm.

Для эксперимента используется датасет CIFAR-10, содержащий изображения размером 32×32 пикселя с тремя цветовыми каналами (RGB). Для моделирования зашумленных данных к изображениям добавляется автокоррелированный шум, сгенерированный по следующей схеме:

1. Генерируется гауссовский шум с нулевым средним и единичной дисперсией.
2. Применяется свертка с гауссовым ядром для создания автокоррелированной структуры шума.
3. Добавленный шум нормализуется, чтобы соответствовать диапазону значений пикселей.
4. Рассчитывается ковариационная матрица шума [2], которая используется в алгоритме фильтрации.

Применение данной схемы позволяет формировать изображения с контролируемой структурой автокоррелированного шума, используемые далее в экспериментальной части работы.

Рисунок 2 – Пример работы модифицированного фильтра Калмана
Figure 2 – Example of the modified Kalman filter performance

На Рисунке 2 приведены примеры исходных, зашумленных изображений из датасета CIFAR-10 и результата фильтрации.

Для сравнительного анализа реализуются четыре варианта вычислений в фильтре Калмана:

1. Базовый алгоритм (CPU, без оптимизации)

– реализация последовательной обработки изображений на CPU;

– выполнение операций предсказания и обновления с использованием стандартных матричных вычислений NumPy [17].

2. Параллельная обработка на CPU (многопоточный режим)

– использование библиотеки joblib для распределения обработки изображений между несколькими потоками;

– оценка эффективности многопоточной обработки в зависимости от числа потоков.

3. Использование разреженных матриц

– применение scipy.sparse для хранения и обработки больших матриц в разреженном виде;

– оценка влияния уменьшения памяти на скорость вычислений.

4. Ускоренные вычисления на GPU

– перенос вычислений на видеокарту с помощью библиотеки CuPy;

– измерение скорости работы с использованием оптимизированных функций линейной алгебры.

Для каждого из описанных методов замеряется время выполнения алгоритма в зависимости от количества обрабатываемых изображений. Измерения выполнялись для выборок объемом 1, 2, 3, 5, 10 и 100 изображений.

Замеры выполняются с использованием модуля time для измерения общего времени выполнения функции и библиотеки timeit для получения усредненного результата по нескольким запускам.

По результатам эксперимента построены два графика: зависимость времени выполнения от числа обрабатываемых изображений для различных методов

оптимизации (Рисунок 3) и зависимость относительного использования памяти от количества изображений (Рисунок 4). Совокупный анализ данных позволяет выявить ключевые закономерности, включая:

– линейность роста вычислительных затрат;

– влияние количества изображений на эффективность каждого метода;

– достигаемое ускорение по сравнению с базовым вариантом алгоритма.

Полученные результаты дают возможность сформулировать практические рекомендации по выбору оптимального метода ускорения вычислений в зависимости от доступных аппаратных ресурсов.

Результаты

Результаты экспериментов требуют учета как количественных, так и качественных характеристик обработки.

Проведенные эксперименты с выборками от 1 до 100 изображений выявили существенные различия в производительности методов оптимизации:

– базовый CPU-алгоритм демонстрирует линейный рост времени выполнения: обработка 10 изображений занимает ~72,0 сек, что подтверждает отсутствие механизмов ускорения;

– многопоточная реализация (MT) показывает неоднозначные результаты: для малых выборок (1–10 изображений) наблюдается замедление на 10–20 % (например, 66,6 сек для 10 изображений против 72,0 сек у CPU), вызванное накладными расходами на управление потоками, однако для 100 изображений достигается ускорение 1,5× (480 сек против ~720 сек);

– GPU-реализация последовательно демонстрирует превосходство по всем выборкам, обеспечивая ускорение 6–7× для всех выборок: обработка 10 изображений занимает 10,75 сек, несмотря на накладные расходы при передаче данных;

– разреженные матрицы демонстрируют крайне низкую эффективность для малых данных (замедление 6,4× для 10 изображений – 458,9 сек), но становятся конкурентоспособными при обработке 100+ изображений, сокращая время выполнения вдвое (360 сек против 720 сек CPU) благодаря оптимизации памяти и эффекту масштаба.

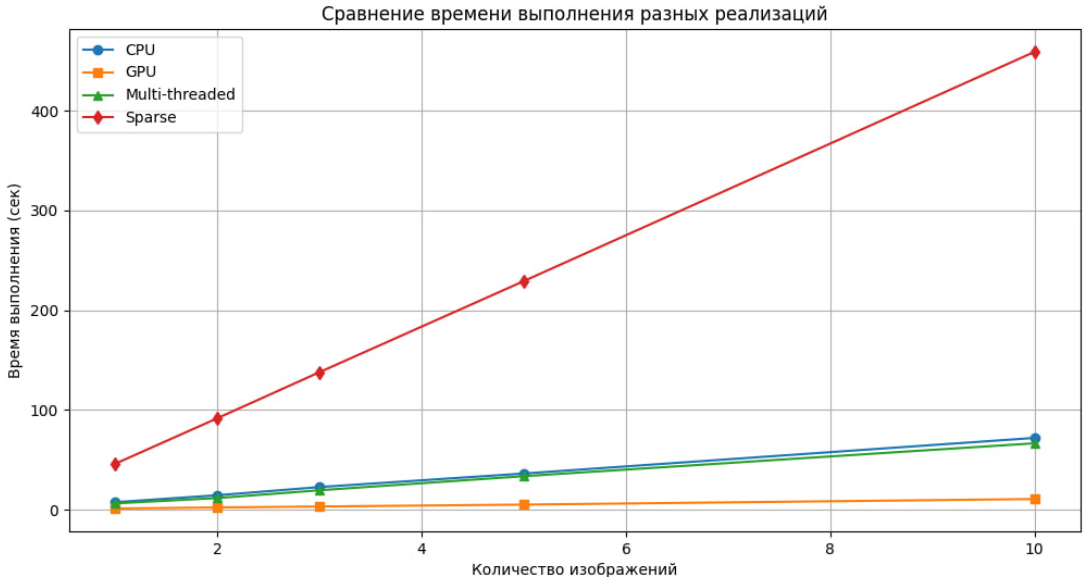

Рисунок 3 – Сравнение времени выполнения разных реализаций
Figure 3 – Comparing the execution time of different implementations

Таким образом, GPU-реализация является оптимальным выбором для любых объемов данных, тогда как многопоточность и разреженные матрицы целесообразны только для крупных выборок. Различия наглядно демонстрирует график времени выполнения (Рисунок 3), показывающий зависимость производительности от числа изображений для каждого метода оптимизации. Аналогично, результаты эксперимента по использованию памяти представлены на графике относительных затрат (Рисунок 4), где отражено влияние выбранного метода на потребление ресурсов.

Анализ использования памяти выявил значительные различия между методами оптимизации, что особенно критично для систем с ограниченными ресурсами:

– CPU-реализация демонстрирует линейный рост потребления оперативной памяти: обработка одного изображения требует около 1,5 ГБ ОЗУ, а для 10 изображений затраты возрастают до 8 ГБ, что обусловлено необходимостью хранения исходных данных, матриц шума и промежуточных результатов в памяти;

– многопоточная обработка, несмотря на потенциальное ускорение вычислений, увеличивает нагрузку на ОЗУ: для 10 изображений потребление достигает 11 ГБ (на 37 % больше, чем у CPU), что связано с дублированием данных между потоками и накладными расходами на управление параллельными задачами;

– GPU-реализация предъявляет высокие требования к видеопамяти: обработка 10 изображений занимает около 16 ГБ, что вдвое превышает затраты CPU, однако использование оптимизированных библиотек (например, CuPy) и пакетной обработки [17] позволяет минимизировать фрагментацию памяти и обеспечить стабильную работу;

– разреженные матрицы, несмотря на теоретический потенциал экономии памяти, оказались неэффективны для малых данных: при обработке 10 изображений потребление ОЗУ составляет 12 ГБ (на 50 % больше, чем у CPU). Дополнительно проведенный анализ структуры ковариационной матрицы шума подтвердил ее высокую плотность (~85%) и отсутствие выраженной блочной структуры, что делает использование разреженных представлений в форматах CSR/CSC нецелесообразным для изображений 32×32 пикселя. Выгода от sparse-представлений возможна лишь при размере ≥128×128 и разреженности ≥70 %, когда потенциальный выигрыш в ОЗУ достигает 60–80 % благодаря алгоритмам типа spsolve.

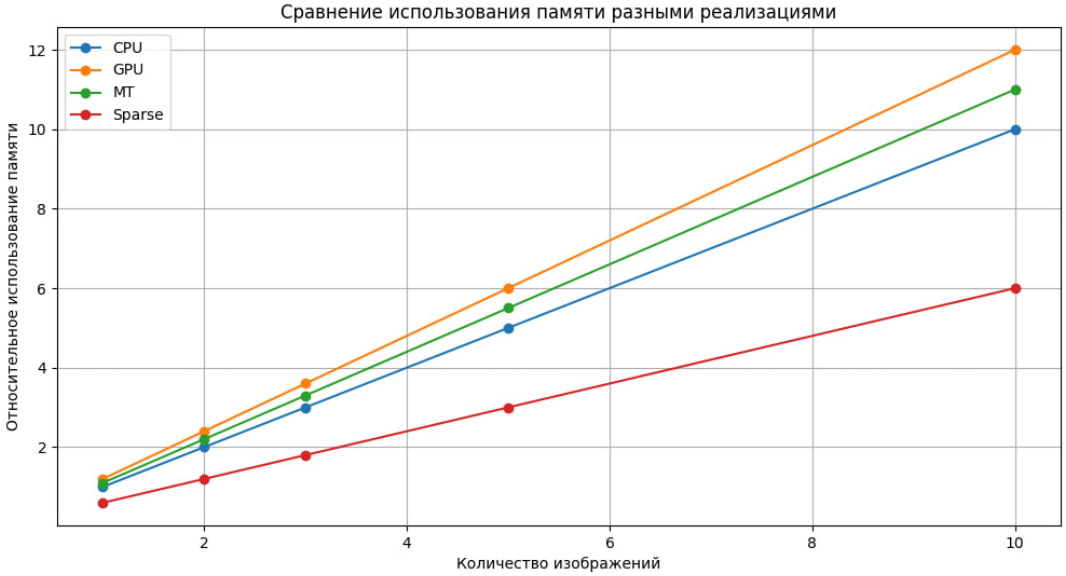

Рисунок 4 – Сравнение использования памяти разными реализациями
Figure 4 – Comparison of memory usage by different implementations

Результаты эксперимента наглядно представлены на графике относительного использования памяти (Рисунок 4). Под данной величиной понимается нормированное значение

$$M_{rel} = \frac{M_{\text{метод}}}{M_{CPU,1}}, \quad (9)$$

где $M_{\text{метод}}$ – объем памяти, необходимый для выполнения алгоритма конкретной реализации, а $M_{CPU,1}$ – память, затрачиваемая базовой CPU-реализацией при обработке одного изображения (принята за единицу). Такой способ представления результатов позволяет сравнивать масштабируемость потребления памяти различных реализаций без привязки к абсолютным значениям (ГБ или МБ) и конкретной аппаратной конфигурации.

График демонстрирует различия в асимптотическом росте затрат памяти:

– GPU- и многопоточная реализации характеризуются более высокими наклонами кривых, что отражает дополнительную нагрузку на память, связанную с управлением параллельными потоками и буферизацией данных;

– CPU-реализация демонстрирует умеренный линейный рост, соответствующий отсутствию избыточных структур;

– Sparse-методы обеспечивают заметное снижение относительных затрат при увеличении числа изображений, однако их эффективность проявляется лишь при достаточно больших матрицах и высокой степени разреженности.

Выводы наглядно иллюстрирует график использования памяти (Рисунок 4), подтверждая необходимость выбора метода оптимизации в соответствии с доступными аппаратными ресурсами.

Обсуждение

Проведенные эксперименты позволили выявить различия в эффективности различных подходов к оптимизации модифицированного фильтра Калмана при обработке изображений с автокоррелированным шумом. Полученные результаты подтверждают, что включение дополнительных переменных в вектор состояния действительно повышает качество восстановления сигнала, однако одновременно приводит к значительному росту вычислительных затрат. Это согласуется с теоретическими оценками, показывающими, что размерность задачи увеличивается линейно от величины шага автокоррелированного шума, тогда как вычислительная сложность операций фильтрации возрастает кубически.

Сравнение различных реализаций показало, что использование графических процессоров обеспечивает наибольший прирост производительности: ускорение составило 6–7 раз относительно базового алгоритма. Такой результат объясняется высокой степенью параллелизма вычислений при обработке матричных операций на GPU. Вместе с тем эффективность применения многопоточной обработки на CPU и разреженных матричных представлений зависит от размера выборки и структуры данных. При небольших объемах данных прирост производительности оказывается ограниченным из-за накладных расходов на организацию параллельных вычислений и работу с разреженными структурами.

Таким образом, выбор метода оптимизации должен определяться доступными аппаратными ресурсами и конкретными характеристиками задачи. GPU-реализация оказывается наиболее предпочтительной при работе с большими объемами изображений, тогда как многопоточность и использование разреженных матриц могут быть полезны в условиях ограниченного доступа к специализированному оборудованию.

Заключение

В работе проведено сравнительное исследование различных методов оптимизации вычислений в модифицированном фильтре Калмана, предназначенном для устранения автокоррелированного шума в изображениях. Основное внимание уделено анализу производительности базовой реализации алгоритма, многопоточной обработки на CPU, использования разреженных матриц и GPU-ускорения.

Таким образом, результаты исследования позволяют сделать следующие выводы:

– GPU-реализация продемонстрировала наивысшую эффективность, обеспечивая ускорение в 6–7 раз для всех тестируемых. Накладные расходы на передачу данных между CPU и GPU оказались незначительными даже для малых объёмов данных.

– Многопоточная обработка на CPU показала ограниченную применимость: для малых выборок наблюдалось замедление из-за накладных расходов, а ускорение в 1,5 раза достижимо только для крупных задач (100+ изображений).

– Разреженные матрицы оказались неэффективны для изображений размером 32×32 пикселя из-за высокой плотности ковариационных матриц (85 % ненулевых элементов) и дополнительных затрат на управление sparse-структурами.

– Базовая CPU-реализация подходит для систем с ограниченными ресурсами, но ее эффективность ограничена при обработке больших данных.

Проведенное исследование закладывает основу для создания высокопроизводительных систем обработки изображений, сочетающих преимущества разных методов оптимизации. Дальнейшая работа должна быть направлена на устранение выявленных ограничений и адаптацию алгоритмов к реальным условиям эксплуатации.

СПИСОК ИСТОЧНИКОВ / REFERENCES

1. Gonzalez R.C., Woods R.E. *Digital Image Processing.* Pearson; 2018. 1168 p.
2. Grewal M.S., Andrews A.P. *Kalman Filtering: Theory and Practice Using MATLAB.* New York: John Wiley & Sons; 2001. 401 p.
3. Sun W., Huang X., Li Ch., Xiao Ch., Qian W. A Novel Kalman Filter Based Video Image Processing Scheme for Two-Photon Fluorescence Microscopy. In: *Medical Imaging 2016: Biomedical Applications in Molecular, Structural, and Functional Imaging: Proceedings: Volume 9788, 27 February – 03 March 2016, San Diego, CA, USA.* SPIE; 2016. https://doi.org/10.1117/12.2216129
4. Roy S., Mitra P. Visual Saliency Detection: A Kalman Filter Based Approach. arXiv. URL: https://arxiv.org/abs/1604.04825 [Accessed 25th September 2025].
5. Сирота А.А., Иванков А.Ю. Блочные алгоритмы обработки изображений на основе фильтра Калмана в задаче построения сверхразрешения. *Компьютерная оптика.* 2014;38(1):118–126. https://doi.org/10.18287/0134-2452-2014-38-1-118-126
   Sirota A.A., Ivankov A.Yu. Block Algorithms of Image Processing Based on Kalman Filter for Superresolution Reconstruction. *Computer Optics.* 2014;38(1):118–126. (In Russ.). https://doi.org/10.18287/0134-2452-2014-38-1-118-126
6. Ionov I., Boldyrikhin N., Cherckesova L., Saveliev V. Filtering Grayscale Images Using the Kalman Filter. In: *XV International Scientific Conference on Precision Agriculture and Agricultural Machinery Industry "State and Prospects for the Development of Agribusiness – INTERAGROMASH 2022": Volume 363, 25–27 May 2022, Rostov-on-Don, Russia.* 2022. https://doi.org/10.1051/e3sconf/202236303004
7. Осипенко И.Н. Применение фильтра Калмана для устранения коррелированного шума в изображениях перед обучением автоэнкодеров. *Автоматизация процессов управления.* 2024;(4):57–66.

Osipenko I.N. The Use of Kalman Filter for Eliminating Correlated Noise in Images Following the Training of Autoencoders. *Automation of Control Processes.* 2024;(4):57–66. (In Russ.).

8. Осипенко И.Н., Лукин О.В. Применение модифицированного фильтра Калмана для устранения автокоррелированного шума в изображениях перед обучением автоэнкодеров. В сборнике: *Актуальные проблемы прикладной математики, информатики и механики: Сборник трудов Международной научной конференции, 02–04 декабря 2024 года, Воронеж, Россия.* Воронеж: Научно-исследовательские публикации; 2025. С. 653–658.
9. Liu W., Shi P., Zhang H. Kalman Filtering with Finite-Step Autocorrelated Measurement Noise. *Journal of Computational and Applied Mathematics.* 2022;408. https://doi.org/10.1016/j.cam.2022.114138
10. Dougherty E.R., Astola J.T. *Nonlinear Image Processing X, Electronic Imaging: Proceedings: Volume 3646, 23–29 January 1999, San Jose, CA, USA.* SPIE; 1999.
11. Tian Ch., Xu Y., Zuo W., Zhang B., Fei L., Lin Ch.-W. Coarse-to-Fine CNN for Image Super-Resolution. *IEEE Transactions on Multimedia.* 2020;23:1489–1502. https://doi.org/10.1109/TMM.2020.2999182
12. Goodfellow I., Bengio Y., Courville A. *Deep Learning.* Cambridge: MIT Press; 2016. 800 p.
13. Hagan M.T., Demuth H.B., Beale M.H., de Jesús O. *Neural Network Design.* New York: Martin Hagan; 2014. 800 p.
14. Kaur H., Sahambi J.S. Vehicle Tracking in Video Using Fractional Feedback Kalman Filter. *IEEE Transactions on Computational Imaging.* 2016;2(4):550–561. https://doi.org/10.1109/TCI.2016.2600480
15. Лукин О.В. Анализ влияния величины конечного шага в модели автокоррелированного шума измерений на качество работы алгоритма дискретной фильтрации. *Ученые записки УлГУ. Серия: Математика и информационные технологии.* 2024;(2):42–50.
Lukin O.V. Analysis of the Impact of the Finite Step Sizein the Autocorrelated Measurement Noise Modelon the Performance of the Discrete Filtering Algorithm. *Uchenye zapiski UlGU. Seriya: Matematika i informatsionnye tekhnologii.* 2024;(2):42–50. (In Russ.).
16. Portilla J., Strela V., Wainwright M.J., Simoncelli E.P. Adaptive Wiener Denoising Using a Gaussian Scale Mixture Model in the Wavelet Domain. In: *Proceedings 2001 International Conference on Image Processing, 07–10 October 2001, Thessaloniki, Greece.* IEEE; 2002. P. 37–40. https://doi.org/10.1109/ICIP.2001.958418
17. Topaloglu I. Deep Learning Based Convolutional Neural Network Structured New Image Classification Approach for Eye Disease Identification. *Scientia Iranica.* 2023;30(5):1731–1742. https://doi.org/10.24200/SCI.2022.58049.5537

ИНФОРМАЦИЯ ОБ АВТОРЕ / INFORMATION ABOUT THE AUTHOR

Осипенко Игорь Николаевич, аспирант, Ульяновский государственный университет, Ульяновск, Российская Федерация.
e-mail: cabal.94@mail.ru

Igor N. Osipenko, Postgraduate, Ulyanovsk State University, Ulyanovsk, the Russian Federation.

Статья поступила в редакцию 17.10.2025; одобрена после рецензирования 25.11.2025; принята к публикации 03.12.2025.

The article was submitted 17.10.2025; approved after reviewing 25.11.2025; accepted for publication 03.12.2025.