Высокопроизводительная реализация метода машинного обучения на основе фрактального сжатия

Е.Ю. Минаев[1,2]

[1]Самарский национальный исследовательский университет им. академика С.П. Королева, Московское шоссе 34А, Самара, Россия, 443086
[2]Институт систем обработки изображений РАН - филиал ФНИЦ «Кристаллография и фотоника» РАН, Молодогвардейская 151, Самара, Россия, 443001

Аннотация. В данной статье исследована параллельная реализация метода машинного обучения с циклическим фрактальным кодированием и использованием словаря доменных блоков, адаптированный для применения на мобильных платформах, с оптимизацией производительности и объема хранимых фрактальных образов изображений. Основная идея метода заключается в применении метода фрактального сжатия на основе систем итерированных функций для понижения размерности исходных изображений, и использовании циклического фрактального кодирования для представления класса изображений в целом. В параллельной реализации метода используется технология CUDA на аппаратной мобильной платформе NVIDIA Jetson Nano, в результате исследований метода получено, что время распознавания в среднем составило 76 мс, что в 2.8 раза быстрее последовательной реализации. Достигнутые показатели производительности являются приемлемыми для использования в системах обработки изображений в реальном времени на мобильных платформах, в т.ч. для БПЛА и наземных автономных роботов.

1. Введение

Проблема использования существующих алгоритмов фрактального сжатия на мобильных программно-аппаратных платформах отмечена в [1]. Традиционно методы фрактального сжатия обладают высокой вычислительной сложностью, и для мобильных платформ не всегда применимы методы и алгоритмы оптимизации производительности разработанные для настольных программно-аппаратных платформ [2]. Современные решения проблемы производительности основаны на применении программируемых пользователем вентильных матриц(FPGA) и использовании графических процессоров[3], что затрудняет использование данных подходов в мобильных платформах. Однако на сегодняшний день становятся доступными и популярными решения на базе мобильной платформы NVIDIA Jetson Nano, обладающей, ко всему прочему, вычислителем с поддержкой CUDA. Исследование параллельной реализации метода машинного обучения на основе фрактального сжатия для данной платформы приведено в настоящей работе. При этом актуальность использования методов фрактального сжатия для мобильных устройств подчеркнута в статье [4].

2. Последовательная реализация машинного обучения на циклическом фрактальном сжатии

Один из перспективных подходов к реализации классификатора на основе фрактального сжатия предложен в [5] При обучении классификаторов описанных в [6], основная проблема заключалась в том, что изображения, формирующие обучающую выборку одного класса, сжимались независимо друг от друга, и объединялись только на этапе построения опорного подпространства. При этом на этапе распознавания возникают проблемы связанные с возможным пересечением опорных подпространств. Соответственно необходимо применять методы обеспечивающие пространственную разделимость, что дополнительно повышает вычислительную сложность. В [7] предложена схема фрактального сжатия использующая несколько различных изображений. В данной статье исследуется классификатор построенный по схеме на замкнутой цепочке изображений из обучающей выборки, с формированием словаря ранговых и доменных областей [8] и соответствующих им преобразований.

Классический алгоритм сжатия на основе систем итерированных функций осуществляет поиск наилучшего аффинного преобразования из доменной области в ранговую. В результате входное изображение кодируется несколькими аффинными преобразованиями:

$$\begin{aligned} \mathbf{I}^* &= F(\mathbf{I}) = \mathbf{C}_{1,4}\mathbf{I} + \mathbf{c}_{5,6}, \\ u^*_{i,j} &= c_7 \cdot u_{i,j} + c_8, \end{aligned} \tag{1}$$

где $\mathbf{I}^* = (i^*, j^*)^T$, $\mathbf{I} = (i, j)^T$ –) координаты пикселей из доменных и ранговых областей соответственно, $\mathbf{C}_{1,4} = \begin{bmatrix} c_1 & c_2 \\ c_3 & c_4 \end{bmatrix}$, $\mathbf{c}_{5,6} = \begin{bmatrix} c_5 \\ c_6 \end{bmatrix}$ – коэффициенты аффинного преобразования, $u^*_{i,j}$, $u_{i,j}$ – значения яркостей пикселей в из доменных и ранговых областях, c_7, c_8 – параметры контраста и сдвига яркости.

В работе используется 8 различных типов преобразований:

$\mathbf{C}_{1,4} = \begin{bmatrix} 0.5 & 0 \\ 0 & 0.5 \end{bmatrix}, \begin{bmatrix} 0 & -0.5 \\ 0.5 & 0 \end{bmatrix}, \begin{bmatrix} -0.5 & 0 \\ 0 & -0.5 \end{bmatrix}, \begin{bmatrix} 0 & 0.5 \\ -0.5 & 0 \end{bmatrix}, \begin{bmatrix} 0.5 & 0 \\ 0 & -0.5 \end{bmatrix}, \begin{bmatrix} -0.5 & 0 \\ 0 & 0.5 \end{bmatrix}, \begin{bmatrix} 0 & 0.5 \\ 0.5 & 0 \end{bmatrix},$
$\begin{bmatrix} 0 & -0.5 \\ -0.5 & 0 \end{bmatrix}$. c_5, c_6 – коэффициенты сдвига и устанавливаются в зависимости от положения блока на изображении. Эти параметры соответствуют различным видам преобразований, таким как вращение, отражение доменных областей с коэффициентом сжатия 0,5, что в соответствии с теоремой Банаха обеспечивает вычислительную сходимость процесса.

Параметры преобразований $c_1 - c_8$ вычисляются в результате фрактального сжатия: $c_1 - c_4$ выбираются из наборов, c_5, c_6 вычисляются в процессе поиска самоподобных блоков, c_7, c_8 – вычисляются по средним значениям яркостей блоков.

Набор преобразований для каждого блока диапазона может быть записан как:

$$I_1 = \bigcup_i F_i(I_0). \tag{2}$$

Используя оператор Хатчинсона, это выражение можно записать в виде:

$$I_1 = FI_0, \tag{3}$$

где I_0 – исходное изображение, F – оператор Хатчинсона, представляющий набор аффинных преобразований, I_1 – результирующее изображение. Схема циклической последовательности преобразований для нескольких изображений обучающего набора представлена на рисунке 1.

После поиска наилучшего аффинного преобразования из доменных областей в ранговые, формируется словарь, включающий информацию о номерах классов изображений, коэффициентах преобразований и схемой деления изображения на ранговые области.

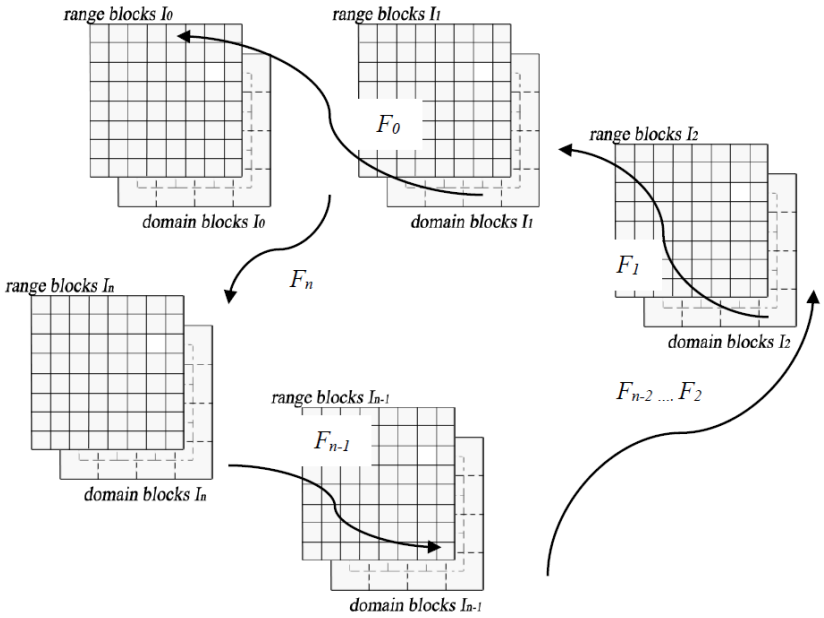


Рисунок 1. Схема циклической последовательности преобразований.

3. Параллельная реализация и экспериментальные исследования

Распараллеливание последовательного алгоритма базируется на двух идеях. Во-первых алгоритм фрактального сжатия допускает распараллеливание по данным, и поиск соответствующих ранговых и доменных областей можно осуществлять независимо как для разных изображений одного класса, так и для разных областей одного и того же изображения. Во-вторых, на этапе распознавания циклическую последовательность преобразований классификатора можно целиком хранить в оперативной памяти графического вычислительного устройства, в т.ч. при ограниченном количестве классов в быстрой разделяемой памяти блоков. В качестве исходных экспериментальных данных была выбрана общеизвестная база радиолокационных изображений MSTAR (moving and stationary target acquisition and recognition). Использовались объекты BMP2, BTR70, T72, для каждого объекта из базы были задействованы обучающие и контрольные выборки. В качестве аппаратной мобильной платформы использовался NVIDIA Jetson Nano. Примеры полученных фрактальных изображений представлены на рисунке 2.

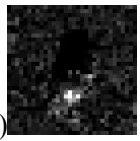
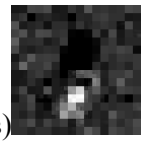
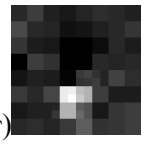
а)  б)  в)  г) 

Рисунок 2. Примеры фрактальных изображений объекта, а – исходное изображение, б,в,г – фрактальное изображение для разбиений ранговых блоков 32×32, 16×16, 8×8 соответственно.

Для верификации параллельной реализации было проведено исследование метода распознавания объектов с циклическим фрактальным кодированием с использованием словаря доменных блоков на трехклассовой классификации, с объектами BMP2, BTR70, T72 (Таблица 1).

Полученные результаты качества распознавания для параллельной реализации метода совпали с последовательной реализацией. Было исследовано быстродействие параллельного метода распознавания на мобильной платформе NVIDIA Jetson Nano (с 128 ядрами CUDA) с сравнении с последовательным методом (на четырехъядерном процессор ARM® Cortex®-A57 MPCore). Получено, что среднее время распознавания для параллельной реализации составляет 76 мс против 214 мс последовательной реализацией, что соответствует скорости обработки в реальном времени.

Таблица 1. Мультиклассовое распознавание для трех типов объектов.

Класс	Доля верно распознанных объектов		
	Последовательный реализация	Параллельная реализация	Saliency Attention and SIFT[9]
BMP2	0.891	0.891	0.64
BTR70	0.882	0.882	0.75
T72	0.904	0.904	0.74

4. Заключение

В результате исследований параллельной реализации метода машинного обучения с циклическим фрактальным кодированием и использованием словаря доменных блоков на базе технологии CUDA на аппаратной мобильной платформе NVIDIA Jetson Nano, получено, что время распознавания в среднем составило 76 мс, что в 2.8 раза быстрее последовательной реализации. Достигнутые показатели производительности являются приемлемыми для использования в системах обработки изображений в реальном времени на мобильных платформах, в т.ч. для БПЛА и наземных автономных роботов.

5. Благодарности

Разработка методов и алгоритмов выполнена при поддержке РФФИ (проект № 17-29-03112-офи-м), экспериментальные исследования - в рамках госзадания ИСОИ РАН - филиал ФНИЦ "Кристаллография и Фотоника" РАН (соглашение № 007-ГЗ/Ч3363/26).

6. Литература

[1] Srivastava, S. Superresolution based Medical Image Compression for Mobile Platforms / S. Srivastava, B. Lall // Workshop on Machine Learning for HealthCare, 2015. – P. hal-01436138.
[2] Chen, D. Fractal video compression in OpenCL: An evaluation of CPUs, GPUs, and FPGAs as acceleration platforms / D. Chen, D. Singh // Design Automation Conference (ASP-DAC), 18th Asia and South Pacific, 2013. – P. 297-304.
[3] Al Sideiri, A. CUDA implementation of fractal image compression / A. Al Sideiri, N. Alzeidi, M. Al Hammoshi, M.S. Chauhan, G. AlFarsi // Journal of Real-Time Image Processing. – 2019. – P. 1-13.
[4] Lima, V. Fast low bit-rate 3D searchless fractal video encoding / V. Lima, W. Schwartz, H. Pedrini // Graphics, Patterns and Images (Sibgrapi), 24th SIBGRAPI Conference, 2011. – P. 189-196.
[5] Minaev, E. Object recognition based on fractal coding using domain blocks // Journal of Physics: Conference Series. – 2018. – Vol. 1096(1). – P. 012099.
[6] Minaev, E. Support subspaces method for fractal images recognition / E.Y. Minaev, V.A. Fursov // CEUR Workshop Proceedings. – 2016. – Vol. 1638. – P. 379-385.
[7] Ozawa, K. Dual fractals // Image and Vision Computing. – 2008. – Vol. 26. – P. 622-631.
[8] Sun, Y. A Novel Fractal Coding Method Based on MJ Sets / Y. Sun , R. Xu , L. Chen, R. Kong, X. Hu // PloS one. – 2014. – Vol. 9(7). – P. e101697.
[9] Karine, A. Saliency attention and sift keypoints combination for automatic target recognition on MSTAR dataset / A. Karine, A. Toumi, A. Khenchaf, M. El Hassouni // Advanced Technologies for Signal and Image Processing (ATSIP), International Conference, 2017. – P. 1-5.

High performance implementation of machine learning method based on fractal compression

E.Y. Minaev[1,2]

[1]Samara National Research University, Moskovskoe Shosse 34A, Samara, Russia, 443086
[2]Image Processing Systems Institute of RAS - Branch of the FSRC "Crystallography and Photonics" RAS, Molodogvardejskaya street 151, Samara, Russia, 443001

Abstract. In this article the parallel implementation of the method of machine learning with cyclic fractal coding and the use of domain block dictionary, adapted for use on mobile platforms, with optimization of performance and volume of stored fractal images is investigated. The main idea of the method is to use the fractal compression method based on systems of iterated functions to lower the dimension of the original images, and to use cyclic fractal coding to represent the class of images as a whole. In the parallel implementation of the method, CUDA technology is used on the NVIDIA Jetson Nano hardware mobile platform, as a result of research of the method, it was found that the recognition time on average was 76 ms, which is 2.8 times faster than sequential implementation. The achieved performance indicators are acceptable for use in real-time image processing systems on mobile platforms, including for UAVs and ground-based autonomous robots.