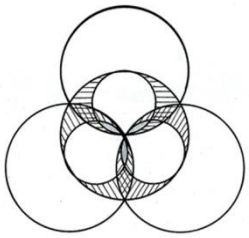
УДК 004.922



ПРИМЕНЕНИЕ МЕТОДОВ АФФИННОГО ПРЕОБРАЗОВАНИЯ МАТРИЦ ЗНАЧЕНИЙ ПИКСЕЛЕЙ РАСТРОВЫХ ИЗОБРАЖЕНИЙ

Ильичев В.Ю. (д.т.н., доц.)

Калужский филиал ФГОУ ВО «Московский государственный технический университет имени Н.Э. Баумана (национальный исследовательский университет)», г. Калуга, РФ; ***patrol8@yandex.ru***

Аннотация. В рамках работы ставились следующие цели: создание метода, алгоритма и программы для сжатия растровой (пиксельной) графической информации с помощью специальных математических приёмов – аффинных преобразований. Основной задачей было обеспечение высокой степени сжатия изображений при минимальном ухудшении их качества. Разработан оригинальный метод замены большого количества пиксельных блоков исходного изображения на относительно небольшое количество наиболее подходящих специально создаваемых доменных блоков. Аффинное преобразование заключается в перемещении любого доменного блока из набора в любую часть изображения, при этом должно обеспечиваться максимальное подобие исходных и доменных блоков. Для осуществления метода разработан алгоритм и программа на современном популярном языке Python. Рассмотрен пример преобразования изображения в оттенках серого размером 256x256 пикселей с применением доменных блоков, созданных из областей изображения размером 4x4 пикселя. В результате получено изображение, визуально не отличающееся от исходного, для описания которого требуется всего 0,3125 информации от исходной. Произведены вычисления и с меньшим количеством доменных блоков. Разработанный метод и программа доказали высокую степень сжатия растровых изображений при сохранении их качества. Возможно дальнейшее совершенствование описанного алгоритма и представленной на сайте автора программы путём одновременного применения разных типов аффинных преобразований. Показано, что тот же метод может быть использован не только для обработки изображений, но также и для выявления подобия (фрактальных свойств) в любом потоке информации.

Ключевые слова: растровая графика, аффинные преобразования, сжатие изображений, фрактальные свойства, язык Python.

1. ВВЕДЕНИЕ И ФОРМУЛИРОВАНИЕ ЦЕЛИ ИССЛЕДОВАНИЯ

Всё увеличивающийся объём передаваемых по различным каналам связи данных требует совершенствования способов их сжатия.

Сжатие информации разного типа (текстовой, графической, звуковой) выполняется по разным алгоритмам. Для достижения наибольшей степени сжатия часто приходится идти на некоторые потери передаваемых данных. При этом желательно не допускать ухудшения качества «сжатого материала», если же снижение качества неизбежно, то оно должно как можно меньше влиять на восприятие особо значимой части информации.

На настоящее время применяют следующие типы сжатия графических данных без потерь информации [19]: RLE, Хаффмана, LZW и др. К сожалению, все они

характеризуются малой степенью сжатия изображений природных, фотореалистичных объектов.

Наиболее известным методом сжатия изображений с потерями является JPG (правильнее его назвать совокупностью методов, выбираемых в зависимости от необходимой степени сжатия и качества получаемого изображения). При этом используются психофизиологические принципы восприятия изображений человеком – отбрасываются компоненты, наименее ценные для восприятия. Основной недостаток сжатия по алгоритмам JPG – появление специфических неприятных искажений (дополнительных артефактов около линий, символов), из-за так называемого эффекта Гиббса [11]. Особенно этот эффект заметен при обработке чертежей даже при небольшой степени сжатия.

В последние годы быстрыми темпами начали развиваться так называемые фрактальные методы сжатия изображений [13]. Фрактальные методы основаны на подобии многих природных явлений и образов, имеющих разный масштаб. Подробно этот вопрос разбирается в многочисленных публикациях, посвящённых фракталам [4, 6]. При этом, в настоящее время фрактальная теория применяется в очень многих отраслях научных исследований, например, при совершенствовании методов проектировании техники [3], её дизайна [18], при создании новых форм искусства, так как часто она адекватно и точно объясняет строение окружающего нас мира.

Целью данной работы являлась разработка алгоритма и программы для обработки растровых изображений с уменьшением объёма файла, путём применения математических методов, составляющих основу теории фракталов - аффинных преобразований.

Примерами аффинных преобразований являются [1]: движение объекта (сюда можно в частности отнести его сдвиг или поворот на плоскости), растяжение или сжатие по осям координат (изменение формы с применением определённых ограничений), отражение относительно осей симметрии и другие. Все виды аффинных преобразований математически можно реализовать определённым изменением значений элементов матриц [2], содержащих координаты точек, значения яркости, цвета и прочие характеристики изображения.

Для сжатия изображений пытаются применять различные виды аффинных преобразований [20] с получением результатов разного качества.

В данной работе применён один из методов аффинного преобразования в оригинальной реализации с помощью специально разработанной программы. Идея данного метода появилась после изучения книги [7].

Его суть состоит в замене более мелких блоков изображения более крупными таким образом, чтобы добиться следующих результатов:

1. минимального изменения исходного изображения при достаточно высокой степени сжатия;

2. узнаваемости исходного изображения при максимальной степени сжатия.

При этом ставились задачи достижения простоты и наглядности алгоритмов сжатия (компрессии) и восстановления (декомпрессии) изображения, а также доступности восприятия программного кода с целью дальнейшего его совершенствования всеми желающими (учёными, программистами, студентами и т.д.).

2. МАТЕРИАЛ И МЕТОДЫ ИССЛЕДОВАНИЯ

Для реализации разработанного метода необходимо произвести ряд преобразований исходного изображения.

Вначале исходное изображение разбивается на крупные пиксельные квадратные блоки одинакового размера, как изображено на рис. 1

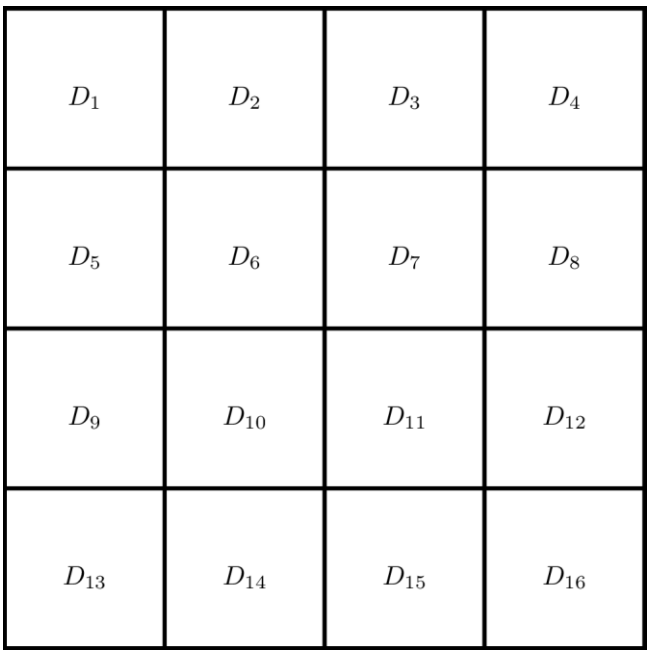


Рис. 1. *Разбиение изображения на крупные блоки (доменные).*

Назовём данные блоки доменными. Они представляют собой матрицы, состоящие из значений пикселей, которые используются для формирования сжатого изображения. При увеличении размера доменных блоков в каждый из них попадает большее количество пикселей исходного изображения. Однако, при этом необходимо выполнение двух условий: все доменные блоки должны иметь одинаковый размер и количество пикселей, составляющих каждую их сторону, должно быть кратным двум.

Затем то же исходное изображение разбивается на мелкие блоки размером 2x2 пикселя (рис. 2).

Данные матрицы значений пикселей назовём конечными. Количество конечных блоков равно 1/4 от количества пикселей исходного изображения (так как размер каждого блока составляет 2x2 пикселя).

Сжатие (уменьшение размера файла) изображения состоит в том, чтобы заменить большее количество конечных блоков на меньшее количество доменных. Такая замена является одним из вариантов аффинного преобразования, так как заключается в перемещении любого доменного блока в любую часть изображения [12]. Естественно, замена должна быть не произвольной, а такой, чтобы образ доменного блока как можно точнее повторял образ конечного. Тем самым будет обеспечено максимальное подобие исходного и сжатого изображений.

R_1	R_2	R_3	R_4	R_5	R_6	R_7	R_8
R_9	R_{10}	R_{11}	R_{12}	R_{13}	R_{14}	R_{15}	R_{16}
R_{17}	R_{18}	R_{19}	R_{20}	R_{21}	R_{22}	R_{23}	R_{24}
R_{25}	R_{26}	R_{27}	R_{28}	R_{29}	R_{30}	R_{31}	R_{32}
R_{33}	R_{34}	R_{35}	R_{36}	R_{37}	R_{38}	R_{39}	R_{40}
R_{41}	R_{42}	R_{43}	R_{44}	R_{45}	R_{46}	R_{47}	R_{48}
R_{49}	R_{50}	R_{51}	R_{52}	R_{53}	R_{54}	R_{55}	R_{56}
R_{57}	R_{58}	R_{59}	R_{60}	R_{61}	R_{62}	R_{63}	R_{64}

Рис. 2. *Разбиение изображения на мелкие блоки (конечные) размером 2x2 пикселя.*

Как видно из рис. 1, доменные блоки имеют большее количество пикселей, чем конечные блоки. Поэтому каждый доменный блок необходимо уменьшить до размера 2x2 пикселя и усреднить значения входящих в него пикселей.

Для восстановления сжатого изображения необходим массив доменных блоков и последовательно расположенные номера элементов этого массива, заменяющих блоки сжимаемого изображения.

Описанный метод предполагает огромное количество расчётов, производимых путём последовательного перебора и сравнения описанных конечных и доменных блоков. Без применения математических и программных методов осуществить данную идею не представляется возможным.

Для программной реализации метода был выбран современный широко и свободно распространяемый язык программирования Python [9], отличающийся простотой программных кодов и универсальностью применения.

Так как растровое изображение можно представить в виде матрицы значений пикселей, которые необходимо преобразовать определённым образом, то существенным фактором, повлиявшим на выбор языка Python, является то, что для него имеется мощная библиотека работы с матрицами Numpy, оптимизированная для достижения высокой скорости расчётов. Данная библиотека позволяет менять вид матриц (разбивать их на блоки или соединять блоки в единую матрицу), производить математические операции над значениями матрицы (например, находить среднее значений элементов массива, их сумму) и многое другое.

Для Python имеется ещё одна необходимая для решения поставленных задач библиотека Matplotlib.image, позволяющая преобразовывать (конвертировать) изображения в матрицы и наоборот [8].

Конечно, для такого большого количества преобразований необходимы навыки работы с большими массивами данных; также пользователю должны быть знакомы

правила правильного выполнения последовательности вычислений во вложенных циклах, понятие рекурсии [21]. Следует отметить, что в данном случае необходимо ещё и образное мышление, так как работа ведётся с многомерными массивами, и необходимо чётко представлять в какой размерности вычисляется каждый параметр.

После разработки методики сжатия изображений с применением аффинных преобразований, необходимо составить алгоритм выполнения программы в выбранной среде Python.

Алгоритм состоит из следующих элементов:

1. Модуль импорта библиотек команд Numpy и Matplotlib.image;

2. Созданная функция outsample для преобразования массива отдельных пикселей в массив из блоков 2x2;

3. Созданная функция insample для преобразования блоков 2x2 в массив отдельных пикселей;

4. Функция конвертации исходного изображения (для примера взято изображение размером 256x256 пикселей в оттенках серого) в массив значений пикселей;

4. Команда создания пустого массива для размещения пикселей доменных блоков (в примере его размерность составляет 64x64 пикселя);

5. Для заполнения доменного массива размерность исходного изображения 256x256 пикселей уменьшается в 4 раза по каждой оси, с усреднением значений пикселей в блоках 4x4 (каждый такой блок заменяется одним пикселем). В данной функции для создания доменного массива пришлось применить рекурсию, т.е. использовать внутри функции саму эту функцию.

6. Применение функции outsample к массиву значений пикселей исходного изображения 256x256 и к доменному массиву.

После всех указанных операций получаем конечный массив 256x256, разбитый на блоки 2x2 и доменный массив 64x64, также разбитый на блоки 2x2.

7. Организуются два вложенных цикла таким образом, чтобы из набора доменных блоков подобрать для каждого из конечных блоков наиболее похожий, путём нахождения наименьшего среднеквадратичного отклонения значений 4 пикселей, составляющих блоки. Таким образом, для того, чтобы найти наиболее похожий пиксель для каждого из 128x128=16384 конечных блоков, необходимо перебрать все из 32x32=1024 доменных блоков.

8. Последний этап – обратное преобразование массива сжатого изображения из блоков 2x2 в массив пикселей с помощью функции insample.

Код созданной программы размещён на странице сайта автора [10] и доступен для детального изучения программной реализации каждого элемента описанного алгоритма.

На рис. 3 в качестве примера представлено исходное растровое изображение размером 256x256 пикселей в оттенках серого. Выбрано изображение природных объектов с множеством деталей, чтобы точнее отследить влияние сжатия на его качество.

Рис. 3. *Исходное изображение до применения сжатия.*

В данном случае исходное изображение (массив пикселей) размером 256x256=65536 было заменено доменным массивом размерности 64x64=4096 (размер файла которого в 16 раз меньше) плюс для каждого из 128x128=16384 конечных блоков необходимо было указать номер доменного блока (то есть используемых элементов изображения стало в 65536/16384=4 раза меньше). Размер сжатого файла, таким образом, составляет 1/4+1/16=0,3125 от исходного.

Изображение, полученное в результате сжатия (замены конечных блоков на доменные), в статье не приводится, т.к. визуально невозможно отличить его от исходного изображения (рис. 3).

3. РЕЗУЛЬТАТЫ И ОБСУЖДЕНИЕ

Из рассмотрения результатов применения приведённого алгоритма на множестве разных образцов растровых изображений следует, что сжатое с применением аффинного преобразования (при размере доменного массива 64x64) изображение визуально не отличается от исходного. При этом достигнута достаточно высокая степень сжатия без потери мелких деталей изображения.

При дальнейшем уменьшении размеров доменного массива качество сжатого изображения начинает ухудшаться. На рис. 4 последовательно показаны изображения с доменными массивами в 32x32=1024, 16x16=256, 8x8=64 блока.

Рис. 4. *Сжатое изображение с доменными массивами 1024, 256, 64 блока.*

В последних двух случаях ухудшение качества изображения становится заметным, хотя объём файла снижается незначительно (так как он в-основном зависит не от размера доменного массива, а от количества конечных блоков изображения, остающегося неизменным). При этом возникае интересный эффект - в целом восприятие изображения не сильно страдает даже при использовании для его создания всего 64 видов «кирпичиков» размером 2x2 пикселя - доменных блоков. Мелкие детали изображения не теряются, оно не выглядит смазанным, нерезким. Кроме того, изображение приобретает «сетчатый» вид, характерный для вышиваемых нитками картин, поэтому рассмотренный приём можно применить как спецэффект при разработке дизайнерских проектов, а также для создания шаблонов для вышивки.

Таким образом, из рассмотрения результатов обработки образцов изображений можно сделать несколько основных заключений:

1. Разработанный метод сжатия с использованием аффинных преобразований растровых изображений в оттенках серого даёт отличные результаты при достижении степени сжатия более 1/3 от объёма исходного файла.

2. При сжатии можно регулировать качество получаемого изображения в широких пределах путём изменения размера доменных блоков, но это не даёт значительного прироста степени сжатия, поэтому представляется нецелесообразным чрезмерно уменьшать количество доменных блоков.

3. Метод позволяет обнаружить фрактальность, самоподобие в любых массивах информации, поэтому может использоваться не только для уменьшения объёма файлов изображений, но и других современных исследований в данном направлении (например, для уменьшения объёма трафика данных [17], при исследовании физических процессов [16], в медицине [15], экономике [14] и т.д.).

4. Описанная техника является простой для реализации средствами современного языка программирования Python.

5. Метод, алгоритм и программу можно совершенствовать, применяя кроме описанного, другие виды аффинных преобразований и формы доменных блоков.

4. ВЫВОДЫ

Таким образом, была полностью выполнена цель данной работы – разработаны оригинальные метод и алгоритм сжатия изображений с применением элементов фрактальной теории, а именно аффинных преобразований. Для реализации алгоритма использован современный язык программирования Python с дополнительными модулями; созданная программа доступна для изучения всеми заинтересованными лицами.

Получены важные результаты научного исследования – доказано, что разработанный метод отличается сравнительной простотой, наглядностью и высокой степенью сжатия изображений без существенной потери важных для восприятия элементов. Благодаря данному методу можно выявлять степень подобия не только изображений, но и наборов данных другого рода, имеющих разный масштаб.

При сжатии изображений по разработанному алгоритму не проявляется эффект Гиббса; более того, - дополнительные исследования, не вошедшие в данную статью,

показали, что применение описанной методики может даже увеличить детальность нерезких изображений.

В целом, работы, нацеленные на совершенствование фрактальных методов [5], должны вывести развитие математического моделирования на новый уровень, предполагающий более точное соответствие моделей реально протекающим в окружающем мире процессам разного рода.

ЛИТЕРАТУРА

1. Абилмажинова Б.С., Андреев В.О. Аффинное преобразование координат или как работает дополненная реальность. // Инновации в науке. – 2016. – № 9 (58). – С. 7-12.

2. Бахрушина Г.И. Моделирование геометрических атак на основе аффинных преобразований. // Учёные заметки ТОГУ. -–2013. – Т. 4. – № 4. – С. 1291-1297.

3. Венгринович В.Л., Рябцев В.Н. Идентификация повреждений методом подвижного фрактала при автоматизированном мониторинге. // Строительство уникальных зданий и сооружений. – 2018. – № 5 (68). – С. 52-61.

4. Дмитриев В.Л., Мухаметова А.К. Популярно о фракталах: Применение фракталов и обзор программ. // NovaInfo.Ru. – 2015. – Т. 1. – № 38. – С. 64-73.

5. Дроботов А.С., Садовникова Н.П. Исследование возможности применения фракталов для решения задач нелинейной оптимизации. // Известия Волгоградского государственного технического университета. – 2008. – № 8 (46). – С. 28-30.

6. Иванов В.В Детерминистические фракталы на основе итерационной последовательности точек в 2D пространстве. // Международный научно-исследовательский журнал. – 2013.– № 7-1 (14). – С. 28-30.

7. Ильичев В.Ю. Разработка программных средств увеличения изображений с использованием их фрактальных свойств. // Системный администратор. – 2021. – № 1-2 (218-219). – С. 124-127.

8. Ильичев В.Ю., Гридчин Н.В. Визуализация масштабируемых 3D-моделей с помощью модуля Matplotlib для Python. // Системный администратор. 2020. – № 12 (217). – С. 86-89.

9. Ильичев В.Ю., Юрик Е.А. Анализ массивов данных с использованием библиотеки Pandas для Python. // Научное обозрение. Технические науки. – 2020. – № 4. – С. 41-45.

10. Код программы сжатия растрового изображения с применением аффинных преобразований. – URL: http://turbopython.ru/prog/Size_decrease_64.py (Дата обращения 14.04.2021).

11. Крылов А.С., Умнов А.В. Влияние эффекта Гиббса на взаимную согласованность в методе разреженных представлений для изображений. // Вестник Московского университета. Серия 15: Вычислительная математика и кибернетика. – 2016. – № 4. – С. 10-15.

12. Кудрина М.А., Мурзин А.В. Аффинные преобразования объектов в компьютерной графике. // Труды международного симпозиума Надежность и качество. – 2014. – Т. 1. – С. 307-310.

13. Лабинский А.Ю. Особенности фрактального сжатия изображений. // Природные и техногенные риски (физико-математические и прикладные аспекты). – 2018. – № 2 (26). – С. 5-12.

14. Лагун А.В. Фракталы в экономике. // Образование и наука без границ: социально-гуманитарные науки. – 2015. – № 2. – С. 307-309.

15. Молчатский С.Л. Топологическая и динамическая характеристики нейронных ансамблей мозга как перколирующих фрактальных множеств. // В мире научных открытий. – 2017. – Т. 9. – № 4. – С. 96-105.

16. Потапов А.А. Турбулентность, фракталы и волны. // В книге: Турбулентность, динамика атмосферы и климата. Международная конференция, посвященная столетию со дня рождения академика Александра Михайловича Обухова. Сборник тезисов докладов. – 2018. – С. 211.

17. Репин Д.С., Филаретов Г.Ф., Червова А.А. Исследование фрактальных характеристик сетевого трафика. // Информатизация образования и науки. – 2019. – № 2 (42). – С. 48-67.

18. Семенихин Д.В. Применение фракталов в дизайне изделий. // Наука в центральной России. – 2013. – № 5S. – С. 17-22.

19. Тулякова М.С. Методы и алгоритмы сжатия графической информации. // Научно-технический вестник Санкт-Петербургского государственного университета информационных технологий, механики и оптики. – 2006. – № 29. – С. 141-144.

20. Усков И.Ю., Нехаев И.Н. Применение аффинных преобразований для трансформации изображений. // Научному прогрессу – творчество молодых. – 2016. – № 3. – С. 134-136.

21. Шестаков А.П. Программирование в профильном обучении информатике: Рекурсия. // Информатика и образование. – 2008. – № 11. – С. 29-32.

APPLICATION OF METHODS OF AFFINITY TRANSFORMATION OF MATRIXES OF RASTER IMAGE PIXEL VALUES

Ilichev V.Y. (Cand. Sci. (Engineering), Associate Prof.)

Kaluga Branch of Bauman Moscow State Technical University, Kaluga, Russian Federation;

patrol8@yandex.ru

Abstract. The goals of the work were: the creation of a method, algorithm and program for compressing bitmap (pixel) graphic information using special mathematical techniques - affine transformations. The main task was to ensure a high compression ratio of images with a minimum deterioration in their quality. An original method has been developed to replace a large number of pixel blocks of the original image with a relatively small number of the most suitable specially created domain blocks. The affine transformation consists in moving any domain block from the set to any part of the image, while maximizing the similarity of the source and domain blocks. To implement the method, an algorithm and program have been developed in the modern popular language Python. An example of converting a 256x256 pixel grayscale image using domain blocks created from 4x4 pixel image regions is discussed. As a result, an image is obtained that is not visually different from the original one, the description of which requires only 0.3125 information from the original one. Calculations have also been made with fewer number of domain blocks. The developed method and program proved a high degree of compression of bitmap images while maintaining their quality. It is possible to further improve the described algorithm and the program presented on the author's website by simultaneously applying different types of affine transformations. It is shown that the same method can be used not only for image processing, but also for detecting similarity (fractal properties) in any stream of information.

Key words: bitmap graphics, affine transformations, image compression, fractal properties, Python language.

REFERENCES

1. Abilmazhinova B.S., Andreev V.O. Affinnoe preobrazovanie koordinat ili kak rabotaet dopolnennaya real'nost' [Affine transformation of coordinates or how augmented reality works]. *Innovations in science*. 2016/ No. 9 (58). P. 7-12.

2. Bakhrushina G.I. Modelirovanie geometricheskikh atak na osnove affinnykh preobrazovaniy [Modeling geometric attacks based on affine transformations]. *Scientific notes of TOGU*. 2013. No. 4-4. P. 1291-1297.

3. Vengrinovich V.L., Ryabcev V.N. Identifikatsiya povrezhdeniy metodom podvizhnogo fraktala pri avtomatizirovannom monitoringe [Identification of damages by mobile fractal during automated monitoring]. *Construction of unique buildings and structures*. 2018. No. 5 (68). P. 52-61.

4. Dmitriev V.L., Mukhametova A.K. Populyarno o fraktalakh: Primenenie fraktalov i obzor programm [Popular about fractals: The use of fractals and a review of programs]. *NovaInfo.Ru*. 2015. No. 1-38. P. 64-73.

5. Drobotov A.S., Sadovnikova N.P. Issledovanie vozmozhnosti primeneniya fraktalov dlya resheniya zadach nelineynoy optimizatsii [Study of the possibility of using fractals to solve nonlinear optimization problems]. *News of Volgograd State Technical University*. 2008. No. 8 (46). P. 28-30.

6. Ivanov V.B. Deterministicheskie fraktaly na osnove iteratsionnoy posledovatel'nosti tochek v 2D prostranstve [Deterministic fractals based on the iterative sequence of points in 2D space]. *International Research Journal*. 2013. No. 7-1 (14). P. 28-30.

7. Ilichev V.Y. Razrabotka programmnykh sredstv uvelicheniya izobrazheniy s ispol'zovaniem ikh fraktal'nykh svoystv [Development of software tools to enlarge images using their fractal properties]. *System administrator*. 2021. No. 1-2 (218-219). P. 124-127.

8. Ilichev V.Y., Gridchin N.V. Vizualizatsiya masshtabiruemykh 3D-modeley s pomoshch'yu modulya Matplotlib dlya Python [Visualization of scalable 3D models using the Matplotlib module for Python]. *System administrator*. 2020. No. 12 (217). P. 86-89.

9. Ilichev V.Y., Yurik E.A. Analiz massivov dannykh s ispol'zovaniem biblioteki Pandas dlya Python [Analysis of data arrays using the Pandas library for Python]. *Scientific review. Technical sciences*. 2020. No. 4. P. 41-45.

10. Kod programmy szhatiya rastrovogo izobrazheniya s primeneniem affinnykh preobrazovaniy [Code of the bitmap compression program using affine transformations. URL: http://turbopython.ru/prog/Size_decrease_64.py (accessed on 14.04.2021).

11. Krylov A.S., Umnov A.V. Vliyanie effekta Gibbsa na vzaimnuyu soglasovannost' v metode razrezhennykh predstavleniy dlya izobrazheniy [The Gibbs effect on mutual consistency in the method of sparse representations for images]. *Bulletin of Moscow University. Series 15: Computational mathematics and cybernetics*. 2016. No. 4. P. 10-15.

12. Kudrina M.A., Murzin A.V. Affinnye preobrazovaniya ob'ektov v komp'yuternoy grafike [Affine transformations of objects in computer graphics]. *Proceedings of the international symposium Reliability and quality*. 2014. No. 1. P. 307-310.

13. Labinsky A.Y. Osobennosti fraktal'nogo szhatiya izobrazheniy [Features of fractal image compression]. *Natural and technogenic risks (physical, mathematical and applied aspects)*. 2018. No. 2 (26). P. 5-12.

14. Lagun A.V. Fraktaly v ekonomike [Fractals in economics]. *Education and science without borders: social and humanities*. 2015. No. 2. P. 307-309.

15. Molchatsky S.L. Topologicheskaya i dinamicheskaya kharakteristiki neyronnykh ansambley mozga kak perkoliruyushchikh fraktal'nykh mnozhestv [Topological and dynamic characteristics of neural ensembles of the brain as percolating fractal sets]. *In the world of scientific discoveries*. 2017. No. 9-4. P. 96-105.

16. Potapov A.A. Turbulentnost', fraktaly i volny [Turbulence, fractals and waves]. *In the book: Turbulence, dynamics of the atmosphere and climate. International conference dedicated to the centenary of the birth of academician Alexander Mikhailovich Obukhov. Collection of abstracts of reports*. 2018. P. 211.

17. Repin D.S., Filaretov G.F., Chervova A.A. Issledovanie fraktal'nykh kharakteristik setevogo trafika [Study of fractal characteristics of network traffic]. *Informatization of education and science*. 2019. No. 2 (42). P. 48-67.

18. Semenikhin D.V. Primenenie fraktalov v dizayne izdeliy [Application of fractals in product design]. *Science in central Russia*..2013. No. 5S. P. 17-22.

19. Tulyakova M.S. Metody i algoritmy szhatiya graficheskoy informatsii [Methods and algorithms for compressing graphic information]. *Scientific and Technical Bulletin of St. Petersburg State University of Information Technologies, Mechanics and Optics*. 2006. No. 29. P. 141-144.

20. Uskov I.Y., Nekhaev I.N. Primenenie affinnykh preobrazovaniy dlya transformatsii izobrazheniy [The application of affine transformations for the transformation of images]. *Scientific progress - the creativity of the young*. 2016. No. 3. P. 134-136.

21. Shestakov A.P. Programmirovanie v profil'nom obuchenii informatike: Rekursiya [Programming in specialized training in computer science: Recursion]. *Informatics and education*. 2008. No. 11. P. 29-32.