10.7256/2305-6061.2013.4.11091

§ 6 КОМПЬЮТЕРНАЯ ГРАФИКА, ОБРАБОТКА ИЗОБРАЖЕНИЙ И РАСПОЗНАВАНИЕ ОБРАЗОВ

Сидоркина И.Г., Кудрин П.А.

АЛГОРИТМ ОПРЕДЕЛЕНИЯ МНОЖЕСТВА БЛИЖАЙШИХ ТОЧЕК ДЛЯ РАСПОЗНАВАНИЯ ТРЕХМЕРНЫХ ИЗОБРАЖЕНИЙ

Аннотация: *Представлено решение задачи выбора эффективного алгоритма определения множества ближайших точек для распознавания трехмерных изображений. Следовательно, от того, насколько эффективно реализован алгоритм поиска МБТ, зависит эффективность всего алгоритма распознавания, использующего МБТ в качестве необходимого звена при обработке изображения. Рассмотрен алгоритм определения множества ближайших точек с помощью деления пространства на кубы (АДПК). Проведен анализ алгоритма, получены математические соотношения, характеризующие временную сложность алгоритма. В статье показано решение задачи оценки АДПК, которая состоит в разбиении на элементарные операции и выражение времени выполнения микроопераций через константы для получения порядка сложности и асимптотических соотношений, которые показывают степень роста времени выполнения алгоритма в зависимости от объема входных данных. Приведены оценки порядка временной сложности для двух реализаций АДПК: последовательной и распараллеленной. Приведена распараллеленная реализация алгоритма и получены оценки ее сложности. Произведено сравнение алгоритма с известными аналогами по временной сложности.*

Ключевые слова: *распознавание образов, сложность алгоритма, множества ближайших точек, эффективность алгоритма, трехмерное изображение, графические процессорные устройства, параллельные алгоритмы, динамические структуры данных, точечное распределение, векторное пространство*

Введение.

Задача поиска множества ближайших точек (МБТ), заключающаяся в отыскании N наиболее близких по евклидовой метрике точек, является одной из подзадач распознавания изображений. Следовательно, от того, насколько эффективно реализован алгоритм поиска МБТ, зависит эффективность всего алгоритма распознавания, использующего МБТ в качестве необходимого звена при обработке изображения.

10.7256/2305-6061.2013.4.11091

Существующие алгоритмы поиска МБТ в лучшем случае обладают порядком сложности $O(n\log n)$ и используют динамические структуры данных для хранения информации о точечном распределении[1] . Использование динамических структур допустимо далеко не на всех существующих высокопроизводительных устройствах, к которым относятся графические процессорные устройства (ГПУ).

Поэтому актуальными задачами являются: создание алгоритмов, использующих структуры данных фиксированного размера, приближение порядка сложности алгоритмов поиска МБТ к линейному виду $O(n)$, и создание, таким образом, более быстрых алгоритмов, чем существующие аналоги. Также важно, чтобы алгоритмы обладали высокой степенью распараллеливаемости, поскольку современные вычислительные устройства имеют параллельную архитектуру.

В статье предложен алгоритм деления пространства на кубы (АДПК), который отличается тем, что позволяет добиться порядка сложности $O(n)$, использует статические структуры данных для своей работы и может быть распараллелен и использован для выполнения на ГПУ. Особенностью АДПК является то, что он аппроксимирует поиск МБТ и является эвристическим, это и обеспечивает скорость его работы. АДПК разработан для оперирования в конечномерном метрическом вещественном векторном пространстве. К существенным достоинствам АДПК относится возможность его использования не только на центральных, но и на графических процессорных устройствах. Для доказательства эффективности алгоритма произведена его оценка.

Ключевым понятием при оценке скорости выполнения алгоритма является *временная сложность алгоритма*, которая выражается в количестве элементарных операций, выполняемых на идеализированном компьютере[2].

Цель работы.

В статье показано решение задачи оценки АДПК, которая состоит в разбиении на элементарные операции и выражение времени выполнения микроопераций через

---

1 Местецкий Л.М. Скелет многосвязной многоугольной фигуры. Труды межд. конф. "Графикон-2005". Новосибирск, 2005.; S. Arya, D. M. Mount, Nathan S. Netanyahu. An Optimal Algorithm for Approximate Nearest Neighbor Searching in Fixed Dimensions. Proceedings of the Fifth Annual ACM-SIAM Symposium on Discrete Algorithms, 1994, pp. 573-582.

2 Ахо, Альфред, В., Хопкрофт, Джон, Ульман, Джеффри, Д. Структуры данных и алгоритмы = Data Structures and Algorithms. — Издательский дом «Вильямс», 2000. — С. 384. — ISBN 5-8459-0122-7 (рус.) / ISBN 0-201-00023-7 (англ.). - с.28-29.

10.7256/2305-6061.2013.4.11091

константы для получения порядка сложности и асимптотических соотношений, которые показывают степень роста времени выполнения алгоритма в зависимости от объема входных данных. Приведены оценки порядка временной сложности для двух реализаций АДПК: последовательной и распараллеленной.

На основе полученной оценки алгоритма произведено сравнение с известными аналогами по скорости выполнения. Получены данные экспериментов, показывающих скорость выполнения последовательного и распараллеленного АДПК.

Описание АДПК.

АДПК состоит из двух главных циклов: распределение и поиск. Задача распределения состоит в формировании структуры данных. Точки, поступившие на вход, проходят индексацию. После этого начинается поиск – извлечение МБТ для точки запроса (иногда ее еще называют полюсом). Специфика поставленной задачи поиска МБТ состоит и в том, что поиск МБТ должен быть выполнен для каждой точки сцены. Это означает, что каждая точка сцены выступает в качестве точки запроса.

Индексация состоит в распределении точек по кубам и в организации двойного сопоставления $[\text{Куб}] \rightarrow [\text{Список Точек}]$ и $[\text{Точка}] \rightarrow [\text{Куб}]$. Такое сопоставление необходимо, чтобы получить по идентификатору точки куб, в котором точка находится, и, наоборот, по идентификатору куба получить список точек, которые в нем находятся.

Поиск состоит:

1) в получении индексов-координат куба, в котором лежит точка,
2) определении по индексам куба ближайших соседних кубов,
3) изъятии из соседних кубов списков точек и записи их в выходной список МБТ,
4) расширении границ поиска МБТ, если не найдено необходимое количество точек.

Принцип, лежащий в основе алгоритма: разбиение пространства на кубы, отнесение точек к этим кубам и поиск множества ближайших точек в соседних кубах. Соседние кубы вычисляются простым прибавлением целого смещения к индексам-координатам куба, в котором лежит точка запроса. Например, имеется куб с координатами (5; 2; 7), тогда ближайшими соседями для него окажутся кубы с индексами (4; 2; 7), (6; 2; 7), (5; 1; 7), (5; 3; 7), (5; 2; 6), (5; 2; 8). Следующими по дальности соседями окажутся кубы, отстоящие от заданного на 2 единицы, затем на 3 единицы, и так далее. Поиск работает путем пополнения искомого МБТ точками из куба, в котором лежит точка запроса, и соседних кубов. В случае если при просмотре соседних кубов нужное для МБТ количество

10.7256/2305-6061.2013.4.11091

точек не может быть найдено, происходит расширение области поиска путем увеличения целого смещения на единицу и вовлечении большего количества кубов в процесс поиска. Как только необходимое количество ближайших соседей для всех точек запроса будет обнаружено, поиск прекращается. На выходе в списках у каждой точки сцены находятся сформированные для них МБТ.

Определение координат куба (i; j; k), в котором лежит точка (x; y; z) производится по формуле:

$$\begin{aligned} i &= Z(\, x / a\,) \\ j &= Z(\, y / a\,) \\ k &= Z(\, z / a\,), \end{aligned} \qquad (2)$$

где Z(t) – целая часть от вещественного числа t,

a – длина ребра куба.

В каждом кубе лежит набор точек, количество которых различно. Возможны ситуации, когда в кубе не окажется ни одной точки.

Опишем алгоритм более детально с помощью псевдоязыка программирования[3]:

```
1  var a = размерРебраКуба;
2  var распределение = создатьПустуюТаблицу();

// Распределение
3  for ( var точка in всеТочки )
   {
4     var индексыКуба = (точка.x / a, точка.y / a, точка.z / a);
5     var идентификаторКуба = вычислитьIdКубаПоИндексам( индексыКуба );
6     var списокТочекКуба = получитьСписокТочекКуба( распределение,
идентификаторКуба );

7     точка.Куб = идентификаторКуба;
8     добавитьТочкуВСписок( списокТочекКуба, точка );
   }
// Поиск
9  for ( var точка in всеТочки )
   {
10    var граница = 1;
11    while ( вПределахДопустимыхГраниц( граница ) )
      {
12          var соседниеКубы = получитьСоседниеКубы( распределение, точка,
граница );
13          for ( var idКуба in соседниеКубы )
            {
14                var списокТочекКуба = получитьСписокТочекДляКуба(
распределение, idКуба );
15                for ( var точкаКуба in списокТочекКуба )
```

3 Номерами слева обозначены операции, которые выполняются в алгоритме

```
                  {
16                      if ( отфильтроватьТочку( точка.МБТ, точкаКуба ) )
                        {
                              continue;
                        }
17                      пополнитьМБТ( точка.МБТ, точкаКуба );
                  }
            }
18          if ( необходимоеКоличествоТочекНайдено( точка.МБТ ) )
            {
                  break;
            }
19          увеличитьГраницыПоиска( граница );
      }
   }
```

Цикл

```
9  for ( var точка in всеТочки )
```

выполняет последовательный выбор точки запроса и поиск МБТ для каждой точки сцены.

Операция

```
16    отфильтроватьТочку( точка.МБТ, точкаКуба )
```

проверяет, следует ли помещать точку в МБТ.

А операция

```
17    пополнитьМБТ( точка.МБТ, точкаКуба )
```

производит поиск местоположения для точки и, если необходимое для МБТ количество точек уже присутствует, то выполняет вытеснение из МБТ самого дальнего от точки запроса соседа.

В результате работы АДПК в МБТ каждой точки будет помещен набор ее ближайших соседей.

Получение асимптотических соотношений для описания порядка сложности последовательного АДПК.

Временная сложность алгоритма обозначается функцией $T(n)$, где n – объем входных данных. Временная сложность имеет порядок $O(f(n))$, если существуют такие n_0 и c ($n_0 = const$, $c = const$, $c > 0$, $n_0 > 0$), при которых для всех $n \geq n_0$ верно неравенство $T(n) \leq cf(n)$[4].

4 Ахо, Альфред, В., Хопкрофт, Джон, Ульман, Джеффри, Д. Структуры данных и алгоритмы = Data Structures and Algorithms. — Издательский дом «Вильямс», 2000. — С. 384. — ISBN 5-8459-0122-7 (рус.) / ISBN 0-201-00023-7 (англ.). - с.28-29.

10.7256/2305-6061.2013.4.11091

Требуется определить порядок временной сложности $O(n)$ в зависимости от объема входных данных n.

Исходные данные:

n – количество точек на сцене,

m – количество точек в МБТ, константа,

k – максимальное количество точек в кубе, константа,

b – максимальная граница области поиска, константа.

Также введем функцию $t(x)$, которая показывает время, требуемое для выполнения операции x.

Поскольку время выполнения нашего алгоритма зависит не только от объема входных данных, но и от самих данных, то $T(n)$ определяется как время выполнения *в наихудшем случае*, т.е. максимум от времени выполнения по всем входным данным.

Для АДПК наилучший случай, когда все точки распределены по сцене равномерно. Наихудший пример, когда точки сосредоточены в двух крайних кубах сцены и количество точек в каждом кубе меньше или равно величине m-1, как это показано на рисунке 1 (см. рис.1). В этом случае количество соседних кубов, которые надо проанализировать АДПК для каждой точки запроса будет максимальным.

Проанализируем алгоритм по шагам для наихудшего случая:

1. Инициализация

$t(\{\text{инициализация}\}) = t(\{1\}) + t(\{2\})$

$t(\{1\}) = const$; - время выполнений операции 1 (по псевдокоду алгоритма).

$t(\{2\}) = const$; - время выполнений операции 2 (создание таблицы + присвоение)

Обозначив за $c_1 = t(\{1\}) + t(\{2\}) = const$, получим

$$t(\{\text{инициализация}\}) = c_1 = const \qquad (1)$$

2. Распределение

$t(\{\text{распределение}\}) = n * t(\{\text{тело цикла } 3\})$

$t(\{\text{тело цикла } 3\}) = t(\{4\}) + t(\{5\}) + t(\{6\}) + t(\{7\}) + t(\{8\})$

$t(\{4\})$, $t(\{5\})$, $t(\{6\})$, $t(\{7\})$, $t(\{8\})$ - время выполнения 4, 5, 6, 7, 8 операций, константы.

10.7256/2305-6061.2013.4.11091

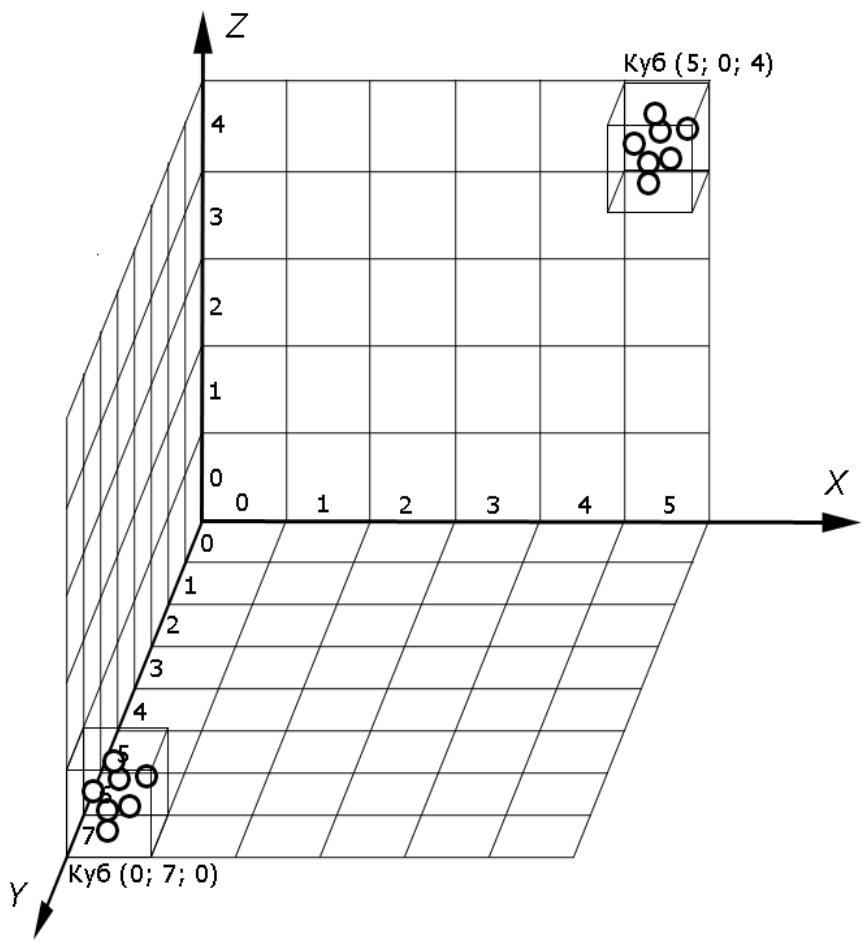

Рис. 1. Наихудший случай.
Кубы (0; 7; 0) и (5; 0; 4) являются крайними, т.е. максимальная координата куба на сцене по оси X равна 5, по оси Y – 7, по оси Z – 4).

Тогда обозначив за $c_2 = t(\{4\}) + t(\{5\}) + t(\{6\}) + t(\{7\}) + t(\{8\}) = const$, получаем

$$t(\,\{\text{тело цикла 3}\}\,) = c_2 = const \tag{2}$$

В таком случае

$$t(\,\{\text{распределение}\}\,) = c_2 n \tag{3}$$

3. Поиск

$t(\,\{\text{поиск}\}\,) = n * t(\{\text{тело цикла 9}\})$,

$t(\,\{\text{тело цикла 9}\}\,) = t(\{10\}) + t(\{\text{цикл 11}\})$,

10.7256/2305-6061.2013.4.11091

$$t(\{\text{цикл } 11\}) = \sum_{r=1}^{b} t(\{12\}) + t(\{\text{цикл } 13\}) + t(\{18\}) + t(\{19\}),$$

$t(\{\text{цикл } 13\}) = P(r) * \big(t(\{14\}) + t(\{\text{цикл } 15\})\big)$, где

$P(r)$ – количество соседних кубов для границы r.

$t(\{\text{цикл } 15\}) = k * (t(\{16\}) + t(\{17\}))$,

$t(\{10\})$, $t(\{12\})$, $t(\{14\})$, $t(\{16\})$, $t(\{17\})$, $t(\{18\})$, $t(\{19\})$ - время выполнения операций 10, 12, 14, 16, 17,18,19, причем значение времени их выполнения константно.

Время выполнения $t(\{16\})$ и $t(\{17\})$ константно, поскольку зависит m, а $m = const$.

Следовательно, сумму $t(\{16\})$ и $t(\{17\})$ можно выразить как функцию, зависящую от m:

$$c_3(m) = t(\{16\}) + t(\{17\}) \tag{4}$$

Получаем:

$t(\{\text{цикл } 15\}) = kc_3(m) = const.$

В таком случае, обозначив за $c_4 = t(\{14\})$, получаем

$t(\{\text{цикл } 13\}) = P(r) * (kc_3(m) + c_4).$

Тогда, обозначив за $c_5 = t(\{12\}) + t(\{18\}) + t(\{19\}) = const$, получаем

$$t(\{\text{цикл } 11\}) = \sum_{r=1}^{b} P(r) * (kc_3(m) + c_4) + c_5 = \sum_{r=1}^{b} P(r) * (kc_3(m) + c_4) + \sum_{r=1}^{b} c_5,$$

Поскольку параметр b фиксирован по условиям задачи и равен некоторой константе, то имеет место равенство

$$\sum_{r=1}^{b} P(r) * (kc_3(m) + c_4) = P(1) * (kc_3(m) + c_4) + P(2) * (kc_3(m) + c_4) + P(3) * (kc_3(m) + c_4) + \cdots + P(b) * (kc_3(m) + c_4) = const,$$

поскольку $P(1)$, $P(2)$, $P(3) \ldots P(b)$ - константы, и, следовательно, каждое слагаемое суммы также равно константе, количество слагаемых ограничено параметром b.

Следовательно, обозначив за

$$c_6 = \sum_{r=1}^{b} c_5 = const,$$

и

$$c_7 = \sum_{r=1}^{b} P(r) * (kc_3(m) + c_4)$$

Получим, что

$$t(\{\text{цикл } 11\}) = c_6 + \sum_{r=1}^{b} P(r) * (kc_3(m) + c_4) = c_6 + c_7 = const.$$

И что,

$$t(\{\text{тело цикла } 9\}) = t(\{10\}) + t(\{\text{цикл } 11\}) = const \qquad (5)$$

Обозначив за $c_8 = t(\{10\})$ и $c_9 = t(\{\text{тело цикла } 9\})$, имеем результирующие соотношения для времени выполнения поиска:

$$t(\{\text{поиск}\}) = n * \left(c_8 + c_6 + \sum_{r=1}^{b} P(r) * (kc_3(\mathrm{m}) + c_4)\right) \qquad (6)$$

$$t(\{\text{поиск}\}) = c_9 n \qquad (7)$$

Для АДПК функция времени выполнения будет выглядеть следующим образом:

$$T(n) = t(\{\text{инициализация}\}) + t(\{\text{распределение}\}) + t(\{\text{поиск}\}) \qquad (8)$$

Выразим функцию $T(n)$ в зависимости от параметров k и b, подставив в (8) формулы (1), (3), (6):

$$T(n) = c_1 + c_2 n + n * \left(c_8 + c_6 + \sum_{r=1}^{b} P(r) * (kc_3(m) + c_4)\right)$$
$$= c_1 + n\left(c_2 + c_8 + c_6 + \sum_{r=1}^{b} P(r) * (kc_3(m) + c_4)\right)$$

Обозначив за $c_{10} = c_2 + c_8 + c_6$, выведем соотношение для $T(n)$:

$$T(n) = c_1 + n\left(c_{10} + \sum_{r=1}^{b} P(r) * (kc_3(\mathrm{m}) + c_4)\right) \qquad (9)$$

10.7256/2305-6061.2013.4.11091

Подставив в (8) выражения из (1), (3), (7), получим
$T(n) = c_1 + c_2 n + c_9 n = c_1 + n(c_2 + c_9)$

обозначив за $c_{11} = c_2 + c_9$, выразим $T(n)$:

$$T(n) = c_1 + c_{10} n \qquad (10)$$

Анализ формулы (10) показывает, что можно найти такие n_0 и c ($n_0 = const$, $c = const$, $c > 0$, $n_0 > 0$), при которых для всех $n \geq n_0$ будет верно неравенство

$$T(n) \leq cn$$

Для параллельного вычислительного устройства выражение (10) примет вид:

$$T(n) = c_1 + \frac{c_{10} n}{z}$$

Анализ формулы (11) показывает, что для идеального параллельного устройства, у которого $z = \infty$:

$$T(n) = const \qquad (12)$$

В результате для идеального параллельного устройства мы можем найти такие n_0 и c ($n_0 = const$, $c = const$, $c > 0$, $n_0 > 0$), при которых для всех $n \geq n_0$ будет верно неравенство

$$T(n) \leq c$$

Таким образом, для идеального параллельного устройства достигается порядок сложности $O(1)$.
На практике же такой результат не достижим, но возможен прирост производительности в $^n/_z$ раз. И чем выше параллелизм устройства, то есть больше значение параметра z, тем быстрее выполняется алгоритм.

Выводы.

В работе предложен алгоритм деления пространства на кубы (АДПК), задачей которого является эффективное и менее требовательное по количеству выполняемых элементарных операций решение одной из важных задач в распознавании образов – поиска множества ближайших точек (МБТ)[5] . Достоинствами предложенного алгоритма являются: малый порядок сложности, составляющий $O(n)$; использование

5 Рябинин К.Б. Решение задачи выбора посадочной площадки беспилотного летательного аппарата на базе кватернионного анализа / К. Б. Рябинин // Вестник МарГТУ. – 2008. – №1(2). – С.33–43.

10.7256/2305-6061.2013.4.11091

статических структур данных для хранения информации о точечном распределении, что делает возможным использование АДПК на высокопроизводительных параллельных вычислительных устройствах, которые не поддерживают структуры, основанные на указателях и динамическом распределении памяти (к таким устройствам, в частности, относятся ГПУ); плюс АДПК обладает хорошей распараллеливаемостью.

Для последовательной и параллельной реализации АДПК получены математические соотношения, показывающие время выполнения и характеризующие порядок сложности, величина которого составила $O(n)$ для последовательной реализации и $O(1)$ для идеального параллельного устройства, в то время как известные аналоги обладают порядком сложности $O(n \log n)$ и даже $O(n^2)$[6]. Таким образом, показывается целесообразность использования алгоритма для поиска МБТ.

Библиография:

1. Ахо, Альфред, В., Хопкрофт, Джон, Ульман, Джеффри, Д. Структуры данных и алгоритмы = Data Structures and Algorithms. — Издательский дом «Вильямс», 2000. — С. 384. — ISBN 5-8459-0122-7 (рус.) / ISBN 0-201-00023-7 (англ.).-с.28-29.
2. Местецкий Л.М. Скелет многосвязной многоугольной фигуры. Труды межд. конф. "Графикон-2005". Новосибирск, 2005.
3. S. Arya, D. M. Mount, Nathan S. Netanyahu. An Optimal Algorithm for Approximate Nearest Neighbor Searching in Fixed Dimensions. Proceedings of the Fifth Annual ACM-SIAM Symposium on Discrete Algorithms, 1994, pp. 573-582.
4. Коробейников А.Г., Кудрин П.А., Сидоркина И.Г. Алгоритм распознавания трехмерных изображений с высокой детализацией. // Вестник Марийского государственного технического университета. Серия: Радиотехнические и инфокоммуникационные системы – Йошкар-Ола: Марийский государственный технический университет – 2010.-№2(9). – С. 91-98.
5. Рябинин К.Б. Решение задачи выбора посадочной площадки беспилотного летательного аппарата на базе кватернионного анализа / К. Б. Рябинин // Вестник МарГТУ. – 2008. – №1(2). – С.33–43.

References:

Akho, Al'fred, V., Khopkroft, Dzhon, Ul'man, Dzheffri, D. Struktury dannykh i algoritmy = Data Structures and Algorithms. — Izdatel'skii dom «Vil'yams», 2000. — S. 384. — ISBN 5-8459-0122-7 (rus.) / ISBN 0-201-00023-7 (angl.).-s.28-29.
2. Mestetskii L.M. Skelet mnogosvyaznoi mnogougol'noi figury. Trudy mezhd. konf. "Grafikon-2005". Novosibirsk, 2005.

6 Рябинин К.Б. Решение задачи выбора посадочной площадки беспилотного летательного аппарата на базе кватернионного анализа / К. Б. Рябинин // Вестник МарГТУ. – 2008. – №1(2). – С.33–43.

10.7256/2305-6061.2013.4.11091

3. S. Arya, D. M. Mount, Nathan S. Netanyahu. An Optimal Algorithm for Approximate Nearest Neighbor Searching in Fixed Dimensions. Proceedings of the Fifth Annual ACM-SIAM Symposium on Discrete Algorithms, 1994, pp. 573-582.
4. Korobeinikov A.G., Kudrin P.A., Sidorkina I.G. Algoritm raspoznavaniya trekhmernykh izobrazhenii s vysokoi detalizatsiei. // Vestnik Mariiskogo gosudarstvennogo tekhnicheskogo universiteta. Seriya: Radiotekhnicheskie i infokommunikatsionnye sistemy – Ioshkar-Ola: Mariiskii gosudarstvennyi tekhnicheskii universitet – 2010.-№2(9). – S. 91-98.
5. Ryabinin K.B. Reshenie zadachi vybora posadochnoi ploshchadki bespilotnogo letatel’nogo apparata na baze kvaternionnogo analiza / K. B. Ryabinin // Vestnik MarGTU. – 2008.-№1(2). -S.33-43.

