

**АКАДЕМИЯ НАУК БССР**  
**Институт технической кибернетики**

**АВТОМАТИЗАЦИЯ ТЕХНИЧЕСКОЙ  
ПОДГОТОВКИ ПРОИЗВОДСТВА**

(направление: "Вычислительная техника в машиностроении")

**научно-технический сборник**

**Выпуск IУ**

**Минск**

**1977**

УДК 681.3.06:61

В.А. Малышев, Г.Н. Плотников

## АЛГОРИТМ ФИЛЬТРАЦИИ ЛИНИЙ ШТРИХОВЫХ ИЗОБРАЖЕНИЙ ПО ТОЛЩИНЕ

Рассматривается одна математическая модель следующей задачи: на изображении имеется несколько линий различной толщины, требуется классифицировать эти линии по толщине, например, отфильтровать их, оставив только линии наименьшей толщины. В рассматриваемой модели указывается алгоритм решения этой задачи и доказывается его оптимальность в смысле надежности и порядка времени его работы.

Штриховые изображения представляют широкий класс объектов, возникающих при обработке информации в ядерной физике, машиностроении, картографии и т.д.

Рассмотрим одну математическую модель следующей задачи: на изображении имеется несколько линий различной толщины, требуется классифицировать эти линии по толщине, например, отфильтровать их, оставив только линии наименьшей толщины.

В формулируемой выше модели указываем алгоритм решения этой задачи и доказываем его оптимальность в смысле надежности и порядка времени работы.

1. Описание модели. Пусть  $\tilde{\Omega} = \{x = (x_1, x_2) : 1 \leq X_i \leq N, i=1,2\}$  — квадрат, а  $\Omega$  — множество его целочисленных точек ( $N^2$  штук). Изображение на  $\Omega$  есть просто функция  $\pi(x)$ , принимающая два значения — 0 и 1. Будем интересоваться классом изображений следующего вида:

$$\pi(x) = (\pi_1(x) \vee \dots \vee \pi_n(x)) \oplus \pi_o(x),$$

где  $\vee$  — дизъюнкция;

$\oplus$  — сумма по  $mod 2$ ;

$\pi_i(x) = 1$  —  $i$ -я линия;

$\pi_o(x)$  — случайная помеха.

Точнее,  $\pi_i(x)$ ,  $i=1, \dots, n$  получается следующим образом. В квадрате  $\Omega$  проводится отрезок прямой или дуга окружности  $L_i$ .  $\pi_i(x)$  полагается равным 1, если находится от  $L_i$  на расстоянии не большем  $\delta_i$  и 0 — в противном случае. Пусть  $\delta_1 > \delta_2 > \dots > \delta_n$ .

Выберем здесь один из простейших вариантов помехи  $\mathbb{X}_0(x)$ . Будем считать ее случайной величиной со значением 0 и 1 в каждой точке  $X \in Q$ , а независимой – в разных точках. Обозначим  $p(x)$  вероятность того, что  $\mathbb{X}_0(x)=1$ , т.е. что точка  $X$  искажается. Уточним  $p(x)$  позже.

**2. Постановка задачи.** Фиксируем некоторое число  $\delta_k > \delta > \delta_{k+1} > 0$ . Требуется найти алгоритм, строящий по  $\mathbb{X}(x)$  изображение  $(\mathbb{X}_1(x), \dots, \mathbb{X}_k(x))$   $\oplus \mathbb{X}'(x)$ , где  $\delta_k > \delta > \delta_{k+1}$ , т.е. отфильтровать все линиитолицы меньшей  $\delta$ .  $\mathbb{X}'(x)$  есть некоторое допустимое отклонение, равное 1 не более чем в  $N < N^2$  точках.

Для того, чтобы иметь возможность сформулировать точные математические результаты об описываемом ниже алгоритме, необходима асимптотическая постановка задачи:

1) общая площадь, занимаемая линиями, мала в сравнении с площадью квадрата. Точнее  $N \rightarrow \infty$ ,  $\frac{N}{n} \rightarrow 0$ ,  $\delta_1 = \text{const}$ ;

2) длина  $L_{ij}$  каждого куска линии  $L_i$ , на котором нет пересечений с этой или другими линиями, сравнима с  $N$ , т.е. существует такая константа  $c > 0$ , что  $\frac{L_{ij}}{N} > c$ . Отсюда следует, что радиус кривизны кривой ограничен сверху числом  $\frac{cN}{2\pi}$  (предполагается, что радиус кривизны каждой линии примерно постоянен). Наш алгоритм работает также для произвольных кривых с ограничением таким образом радиусом кривизны;

3)  $p(x) \neq p$ . От этого предположения можно частично отказаться. При этом  $N^2 p \delta_n \rightarrow 0$ ;

4) каждая точка линии находится либо на расстоянии меньше  $2\delta$  от пересечения этой линии с другой, либо на расстоянии больше  $2\delta$  от любой другой линии.

Алгоритм, описываемый ниже, дает при  $n$  чисто линейного перебора точек из  $Q: X'_1, \dots, X'_n$ , и попутно совершают некоторое другое операции. Под сложностью алгоритма будем понимать обычную алгоритмическую сложность (например [1]). Как будет видно ниже, сложность определяется числом  $M$ . Будет доказано, что для нашего алгоритма  $M \propto N^4$ , что  $M$  не может быть меньше, т.е., приводимый ниже алгоритм оптимальен.  $M \propto N$  означает, что  $0 < \lim_{N \rightarrow \infty} \frac{M}{N} \leq \lim_{N \rightarrow \infty} \frac{M}{N} < \infty$ .

**3. Описание алгоритма.** Рассмотрим прямые в  $\mathbb{E}$  вида

$$X_1 = k \left[ \frac{cN}{\pi \alpha_0} \right], 1 \leq k \leq \frac{\pi \alpha_0}{c} \quad \text{и} \quad X_2 = k \left[ \frac{cN}{\pi \alpha_0} \right], 1 \leq K \leq \frac{\pi \alpha_0}{c}$$

и занумеруем их точки в произвольной удобной для программирования последовательности:  $X'_1, \dots, X'_{c\alpha_0}$ ,  $\alpha_0 = \text{const} + 1$ ,  $\alpha_0 > 1$ . Алгоритм последовательно "попадает" в эти точки. При попадании в точку  $X'$  алгоритм проверяет наличие линии около точки  $X'$  по подпрограмме 1. Если линии нет, то он переходит к точке  $X'^{++}$ . Если линии есть, то алгоритм оценивает ширину линии по подпрограмме 2. Если ширина больше  $\delta$  или имеет место пересечение линий, то алгоритм переходит к точке  $X'^{++}$ .

В противном случае алгоритм начинает отолживать и стирать линию по подпрограмме 3. После окончания работы подпрограммы 3 алгоритм переходит в точку  $x^{i+1}$ , где все начинается сначала. Алгоритм останавливается при попадания в точку  $x^{\frac{2\pi}{c}d_0}$ .

**Подпрограмма 1.** Подсчитывает число  $g$  единиц в квадратном "окне" со стороной  $d_1$ , и с центром в точке  $x^i$ . Если  $g < \alpha, \delta_1, \delta_2, \dots, \delta_k < 1$ , то алгоритм переходит в точку  $x^{i+1}$ . В противном случае алгоритм начинает работать по подпрограмме 2.

**Подпрограмма 2.** Выберем ширину  $d_2$  второго окна так, чтобы внутри него линия могла считаться прямой, т.е.  $d_2$  много меньше нижней границы радиуса кривизны.

**Замечание.** Для окружностей нижняя граница равна  $\frac{cN}{2\pi}$ . Оценка ширины линии производится статистической оценкой отрезков  $a$  и  $b$  (рис. 1)

$$a = \frac{1}{d_2} \sum_{i=1}^{d_2} a_i,$$

где  $a_i$  - число единиц в  $i$ -й строке окна. Аналогично

$$b = \frac{1}{d_2} \sum_{i=1}^{d_2} b_i,$$

где  $b_i$  - число единиц в  $i$ -м столбце окна. Отсюда ширина линии

$$y = \frac{ab}{\sqrt{a^2 + b^2}}.$$

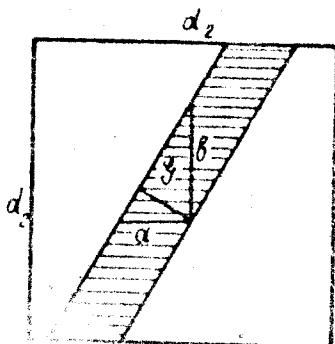


Рис. 1

Для установления наличия пересечения в каждой строке подсчитываются те же величины. Если линия из набора ширин  $\delta_1, \dots, \delta_k$  проходит через окно, то эмпирическое  $y$  будет не меньше  $\delta_k$ . Если  $y > \delta_{k+1} + \frac{\delta_{k+1} - \delta_k}{2}$ , то алгоритм переходит к точке  $x^{i+1}$ . В противном случае начинает работать подпрограмма 3.

Дополнительно тем же способом вычисляется  $y$  для областей  $A_i, i=1, \dots, 4$ , указанных на рис. 2. Если где-либо окажется, что  $y < \delta_n$ , то алгоритм переходит в точку  $x^{i+1}$ . Этим устраняются случаи, указанные на рис. 3.

**Замечание.** В важном для практики случае, когда  $n-k=1$ , т.е. имеются тонкие линии только одного типа, подпрограмма 2 отсортирует все пересечения вообще.

**Подпрограмма 3.** В нижней строке ищется 1. Пусть она в точке  $X$ . От нее на трех отрезках (рис. 4) длины  $l = \max(a, b)$  измеряется число единиц. Следующая точка берется в конце того отрезка, где число единиц максимально.

Далее процедура повторяется с той разницей, что из внутренней точки измерения производится по четырем возможным направлениям отрезкам той же длины (рис. 3). Когда подаем по другой границе в точку  $y_1$ , то строим окно ширины  $d_2$  с центром в этой точке, при чем все повторяется сначала, предварительно стирая все в окне  $A - B$ , где  $A$  - старое окно,  $B$  - новое окно.

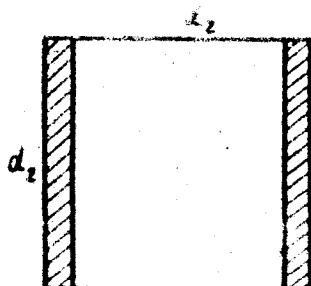


Рис. 2

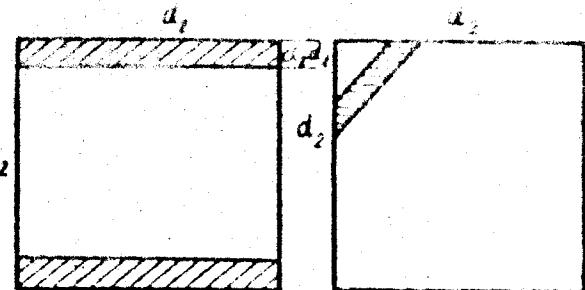
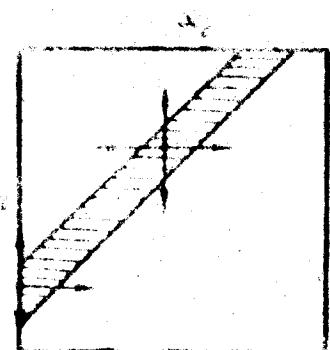
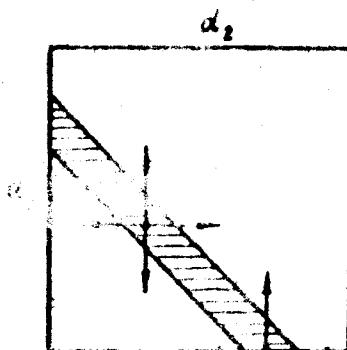


Рис. 3

Далее опять применяем подпрограммы 2 и 3. Если же подпрограмма 2 обнаружит пересечение или линию большей ширины или касательной участок линии, то работает подпрограмма 4. После чего алгоритм переходит к точке  $I^{(k)}$ .



**Подпрограмма 4.** Стирает часть тонкой линии, находящуюся в окне ширины  $d_2$ . Это несущественная часть и здесь же будет ее описание.

#### 4. Результаты.

1. Сложность данного алгоритма  $T \leq K$ .
2. Вероятность того, что  $|X'| \in (X_0 + \text{const})$ , где  $X'$  - это число единиц в изображении  $L$ , стремится к 1.
3. Не существует алгоритма сложности 1, такой что он решает задачу, задающую свойством 2.

Иначе говоря, построенный нами алгоритм оптимален в указанном точном смысле этого слова.

Скажем несколько слов о доказательствах. Предположим, что выбрали  $\tilde{x}$  точек в  $\Omega$  для обследования. Так как линия имеет случайное направление, то вероятность того, что хотя бы одна из точек  $\tilde{x}$  попадает на эту линию, стремится к нулю при  $N \rightarrow \infty$ . Отсюда нетрудно получается утверждение 3.

Утверждение 1 получается стандартным подсчетом сложности и выполнением всех предельных переходов.

Утверждение 2 доказывается сложнее и здесь его доказывать не будем.

Параметрами алгоритма, подлежащими выбору, являются:  $N$  - длина стороны рисунка;  $C$  - априорная нижняя граница длины связных компонент изображения;  $d_0 \approx 2$  - константа, определяющая число сечений;  $d_1$  - ширина скана в подпрограмме 1;  $\alpha_1 \approx \frac{1}{2}$  - константа в подпрограмме 1;  $d_2$  - ширина скана в подпрограмме 2;  $\alpha_2 \approx \frac{1}{2}$  - константа в подпрограмме 3. Предполагается, что данный алгоритм может быть эффективно применен в задачах ввода фильмомой информации с автоматов на ЭЛТ широкого класса штриховых изображений.

### Л и т е р а т у р а

1. Клобус Б.М., Модестова В.А. Оценка сложности некоторых классов функций. - "Вестник МГУ", 1965, вып. 4, с. 44-61.

Москва

①

Алгоритм для решения линейных уравнений по голомицкому

B.A. Малышев, Чеканов Б.А., Адакес В.Б., Курникова Н.,  
Платоновский Г.Н.

Мы рассмотрим один из методов решения линейных уравнений по голомицкому методом следующим образом: на изображении ищется несколько линий параллельных голомицким, требуется классифицировать эти линии по голомицким, например, откладывать их, оставив только линии ~~находящиеся~~ голомицким.

Алгоритм В фрагментированной форме показан на изображении алгоритма решения этой задачи, где в укачивании алгоритма решения этой задачи, где и доказывается его однозначность в линии ~~найденных~~ ~~найденных~~ и ~~найденных~~ ~~найденных~~ и порядок времени его работы.

Описание модели.

1. Изображение модели.

На изображении  $\tilde{S}$  — уравнение  $= \{x = (x_1, x_2) : x_1 \leq x_i \leq N\}$  — уравнение  $S$  — уравнение  $x_1 = 0$  и  $x_2 = 0$ . Изображение на  $\tilde{S}$  — уравнение  $x_1 = 0$  и  $x_2 = 0$ . Изображение на  $S$  — уравнение  $x_1 = 0$  и  $x_2 = 0$ .

Если прямая  $y_i(x)$  на  $S$ , принадлежащая тем же изображениям, то  $y_i(x)$  на  $\tilde{S}$ , принадлежащая тем же изображениям,  $0$  и  $1$ .

Мы будем интересоваться классом изображений следующего вида:

$$y_i(x) = (y_{i1}(x) \dots 1 \dots 1 y_{in}(x)) \oplus y_{i0}(x)$$

где  $1$  — единица,  $\oplus$  — сумма по  $\text{mod } 2$ ,

$y_{i1}(x)$  —  $i$ -таке линия, а  $y_{i0}(x)$  — сдвигнутое копия.

Также,  $y_i(x)$ ,  $i=1, \dots, n$ , показывает следующие обстоятельства. В  $\tilde{S}$  от  $\tilde{x}$  откладывается  $\tilde{r}$  и проводится отрезок прямой или дуги окружности  $L_i$ .

$y_i(x)$  называется рабиной 1, если  $x$  отстоит  $\delta_i$   
от  $L_i$  на расстояние не большее  $\delta_i$ , и 0 —  
в противном случае. Итак  $\delta_1 \geq \delta_2 \geq \dots \geq \delta_n$ .

~~Чтобы убедиться в этом, рассмотрим~~  
Мы выберем здесь один из простейших вариантов  
помехи  $y_0(x)$ . Мы предположим её независимой  
в разных точках, ~~и однозначно определяющей~~  
~~изображение~~  $p(x)$  — ~~точка~~  $x$  искается.  
тако, что  $y_0(x) = 1$  в  $\overset{r_0}{\text{окрестности}}$  точки  $x$  и скрывается.  
Мы увидим  $p(x)$  носке.

## 2. Постановка задачи.

Рассмотрим некоторое число  $\delta > 0$ . ~~Чтобы~~  
Требуется найти алгоритм, строящий по изображению  
 $y(x)$  изображение  $(y_1(x) \dots y_k(x)) \oplus y'(x)$ ,  
где  $\delta_k \geq \delta > \delta_{k+1}$ , т.е. отображать  
всё выше полученные меньшее  $\delta$ .  $y'(x)$  есть  
некоторое допускаемое отображение, рабине 1 не  
более, чем в ~~всех~~  $\overset{N}{\text{точках}}$ .

Для того, чтобы иметь возможность сформулиро-  
вать задачу математически, рассмотрим об-  
общенное выше описание, необходимое  
асимптотическая постановка. Поэтому мы  
 будем предполагать, что (здесь, в то же

рассматривает в логарифмических предположениях, а (3)  
утверждение о его оптимальности ~~доказано~~ доказано  
только в фрагментарных и кратких доказательствах):

1. Относительная, заменяющая линейную, цена  
в сравнении с производством квадрата. Т.е.

$$N \rightarrow \infty, \frac{n}{N} \rightarrow 0 \quad \delta_i = \text{const}$$

каждое кубическое значение  $L_i$ , на котором нет пересечений с другими значениями

2. Данные о производительности сравниваются с  $\frac{N}{\ln N}$ , т.е.

где  $L_{ij}$  — кубик

Существует также константа  $C$ , такая что  $\frac{L_{ij}}{N} > C$

Вставка на обратное \*\*)

3. Амортизация. Используется формула  $\rho(x) = \rho_0 e^{-\lambda x}$ ,  
т.е. имеется линейная изгаряка. От  $x=0$  предполагается что изгаряка отсутствует.  
При этом  $N^2 \rho^{\delta_n} \rightarrow 0$

на обратное \*1)

Амортизация, описываемая выше,ает правило  
расчетного перехода токи из  $\mathcal{R}$ :  $x^1, \dots, x^{m+1}$   
и поэтому совершают некоторые другие обмены.  
под спортивного алгоритма что будем называть  
общим спортивного алгоритмического спортивного  
(ан., например, 11)). Делается как будто  
будет выше, затем же спортивного алгоритма примерно  
одинаковой спортивной примерно одинаковой спортивной примерно  
он предполагает затем же спортивного алгоритма спортивного алгоритма  
также алгоритмический алгоритмический Т.е., будем

алгоритмический алгоритмический Т.е., будем  
доказано, что  $M \leq N$  также называется алгоритмический

и что  $M$  не может быть меньше, т.е. приведенный

выше алгоритм оптимален.

$$*) M \approx N означает что \lim_{n \rightarrow \infty} \frac{M}{n} = \lim_{n \rightarrow \infty} \frac{N}{n} = 1$$

# Кризисы

\*)

4. Каждое тело линии находится либо на расстоянии  $\leq 2\delta$ , от пересечения этой линии с другой либо. на расстоянии  $> 2\delta$ , от любой другой линии

и

---

\*\*) Отсюда следует, что радиус кривизны первичной орбиты меньше чем  $\frac{c}{2\pi}$ .  
Наш алгоритм подбирает также для производимых кривых с ограниченным таким образом радиусом кривизны.

(4)

### 3. Описание алгоритма

Рассмотрим правило в  $\mathbb{R}$  буда:

$$x_1 = K \left\lceil \frac{\frac{c}{d_1} \Delta_1 \alpha_1}{\pi \sqrt{d_1}} \right\rceil, \quad 1 \leq K \leq \frac{\pi \Delta_1 \alpha_1}{c}, \quad x_2 = K \left\lceil \frac{\frac{c}{d_2} \Delta_2 \alpha_2}{\pi \sqrt{d_2}} \right\rceil, \quad 1 \leq K \leq \frac{\pi \Delta_2 \alpha_2}{c}$$

и заменяющим их точки в ~~координатной~~ удобной для приведения ~~форму~~ последовательности:

$$x^1, \dots, x^{\frac{\pi \Delta \alpha}{c} \Delta_0}, \quad \Delta_0 = \text{const.}$$

Алгоритм последовательно "подбрасывает" в эти точки.

При попадании в точку  $x^i$  алгоритм проверяет наличие либо около точки  $x^i$  по подпрограмме 1. Если либо нет, то он переходит к точке  $x^{i+1}$ . Если либо есть, то алгоритм вычисляет ширину либо по подпрограмме 2. Если ширина больше 0 или имеет место пересечение линий, то алгоритм переходит к точке  $x^{i+1}$ . Всё ~~делает~~ вычисление и сперва либо по подпрограмме 3. После окончания работы подпрограммы 3 алгоритм переходит в точку  $x^{i+1}$ , где все начинает сначала.

Алгоритм ~~также~~ останавливается при попадании в точку  $x^{\frac{\pi \Delta \alpha}{c} \Delta_0}$ .

#### Подпрограмма 1.

Подсчитывает число единиц в квадратном "окне" со стороной  $d_1$  и с центром в точке  $x^i$ .

~~Но это не всегда~~ Если  $g \leq d_1 \Delta_1 \alpha_1 / \pi \sqrt{d_1}, \alpha_1 < 1$ , то алгоритм не попадёт в точку  $x^{i+1}$ . В противном случае алгоритм начинает работать по подпрограмме 2. О боязне  $d_1$  и  $d_2$  мы скажем позже.

## Нодография 2.

(5)

Выберем ширину борта окна так, чтобы  
выходы из окна не мешали спускаю Примой,  
т.е.  $d_2 \leq$  радиус огибания внешней границы  
радиуса кривизны (Ось огибающей  
Гипотеза). Для определения эта ширине

ширина борта  $\frac{cN}{2\pi}$ .

~~Проблема~~ ~~помеха~~ ~~окна~~ ~~ширина~~ ~~минимум~~

применимое стационарное  
значение отрезка  $a + b$

При этом

$$a = \frac{1}{d_2} \sum a_i$$

где  $a_i$  — это единица  $b$   $i$ -го сечения окна.

Аналогично  $b = \frac{1}{d_2} \sum b_i$ , где  $b_i$  — это единица  $b$   $i$ -го сечения окна.

Будем ширину  $b$

$$\gamma = \frac{ab}{\sqrt{a^2 + b^2}}$$

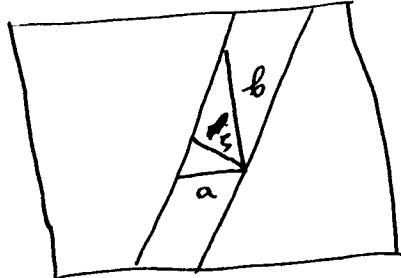


Рис. 1

При этом для уничтожения пакетов пересечений в первом сечении погонажного те же величины радиуса, что и в окне, т.е.  $R = d_2$ ,  
причем  $\delta_1 = \delta_2$  и пакеты отсутствуют

Если ~~ширина~~ ~~ширина~~  $\delta_1, \dots, \delta_k$  проходит  
через окно, то  $\gamma$  будет не меньше  $\delta_k$ .

Если  $\gamma > \delta_{k+1} + \frac{\delta_{k+1} - \delta_k}{2}$ , то алгоритм  
переходит к точке  $x^{k+1}$ . В противном случае  
нашивает рабочий подпрограмма 3.  $\checkmark$  А) Оставая  
на обработке

№1. Дополнительно тем же способом  
показывается что если для  $\delta$  дана областей  
 $A_i$ ,  $i=1, \dots, 4$ , указанных на рис. 2. Если  
 все либо скрещиваются, либо  $\delta \subset \delta_n$ , то аэрофиль  
 переходит в точку  $x^{i+1}$ . Тогда уединённость  
 симметрии, указанное на рис. 5.

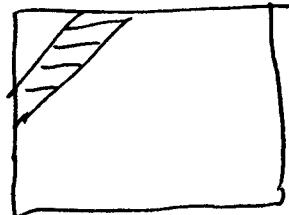


Рис. 5.

(6)

## Метод отражения 3.

Замечание. В вершине дуги  $n-k=1$ , т.е. ~~и неотраженное~~ ~~лучи~~ ~~также~~ отсчитывается от ~~стартовавшего~~ ~~стартует~~ ~~бес~~ ~~непрерывно~~ ~~воздуха~~.

## Метод отражения 3.

~~Прием отраженного сигнала~~ ~~засек единицу времени~~ ~~блока~~ ~~запись~~  
~~в блоки~~  $A_1, A_2, A_3, A_4$ . (см. рис. 2)

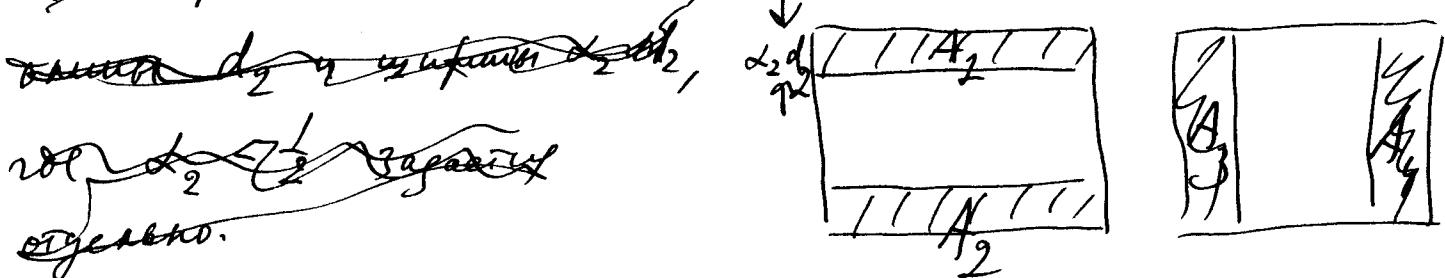


Рис. 2.

## Метод отражения 3

~~последовательное отражение~~ ~~и отражение~~  
~~луча~~ ~~имеет~~ ~~один~~ ~~особенность~~ ~~где~~ ~~луч~~  
~~из блоков~~ ~~имеет~~ ~~бесконечную~~ ~~расстояние~~ ~~до~~ ~~стартовавшего~~ ~~луча~~ ~~в~~ ~~точке~~  $x$ .

В первом случае имеет 1.). А от неё

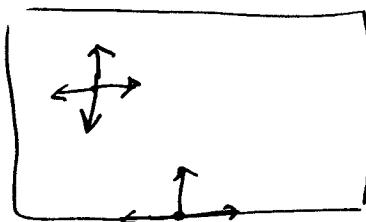
по трем отражениям (рис. 3)

~~имеет~~ ~~длину~~  $l = \max(a, b)$   
~~имеет~~ ~~один~~ ~~единицу~~.

Следующий луч берётся в

также ~~одно~~ ~~отражение~~, ~~один~~ ~~единицу~~ Рис. 3.  
~~имеет~~ ~~один~~ ~~единицу~~.

Данная процедура повторяется с той  
~~разницей~~, что ~~один~~ ~~из~~ ~~внешней~~ ~~горизонта~~



изменение производится по методам  
взаимно перпендикулярных отрезков от спе-  
циальных (рис. 3). Когда движется до другой  
границы  $f$  в точке  $Y_1$ , то сориенти-  
ровка циркуля  $d_2$  с центром в этой точке, ~~и~~  
носит уже новое значение, предвари-  
тельно спроектированное на плоскость  
бесконечности в области  $A-B$ , где  
 $A$  - одно окно,  $B$  - новое окно.

~~Методом~~ Дане описание применения ~~и~~ подгра-  
ничию 2 и 3. Если при подгранице 2  
одновременно ~~имеются~~ <sup>имеются</sup> пересечение или ~~пересечения~~ <sup>одинаковы</sup> границ  
циркуля, то ~~работает~~ <sup>имеет</sup> подграница 4.  
Но если зеро алгоритм переходит к зерву  
 $X^{i+1}$ .

Подграница 4 стартует для зеркальной  
линии, то падающей в окно циркуля  $d_2$   
и то несущей зеркальной ~~линии~~ и ее зеркаль-  
ного биссектрического изображения.

(8)

#### 4. Рекурсивные алгоритмы

1. Сложность гамильтонова алгоритма  
 $L \propto N$ .

2. Вероятность  $\tau_{100,200}$  гамильтонов  
 алгоритм  $\leq \frac{1}{N!} \approx \frac{1}{N^N}$   
 $|h_1| \leq |h_0| + \text{const}$ ,  
 где  $|h_i|$  - число единиц в изображении  $h_i$ ,  
 стремящее к 1.

3. Их существует алгоритм  $L$  такой,  
 что  $\frac{L}{N} \rightarrow 0$ , обладающий свойством 2.  
 Число ребер, построенных этим алгоритмом  
 определяется в указанном выше смысле  
 этого слова.

Следует упомянуть о доказательствах  
 в § 2  
 Предположим, что мы выбрали  $L$  так ~~чтобы~~  
 обследование. Так как число имеет ограниченное  
 наибольшее, то вероятность того, что хотя бы  
 одна из трех  $L$  попадет на эту линию,  
 стремится к нулю при  $N \rightarrow \infty$ . Отсюда  
 легко получаем утверждение 3.

Утверждение 1 получается стандартным  
 методом сокращения и сводится к тому же

пределам переходов.

№ 10 Утверждение 2 доказывает алгоритм  
и мы здесь его доказывать не будем.

### 5. Некоторые практические замечания.

В конкретных случаях алгоритм может быть улучшен за счет оптимизации выбора параметров, пересчитанных выше. Однако ренейтуру их выбора дать трудно. Поэтому мы рассмотрим следующий числовой пример.

## Параметры алгоритма.

## Приложение (1)

$N$  - общее количество рисунков,

$C$  - алгоритмическая метрика граница для длины

$d_0$  - константа, определяющая радиус сферы —  
 ~~$d_1$~~  — ширине ~~одной~~ окна в подпрограмме 1

$d_1$  — константа в подпрограмме 1.

$d_2$  — ширине окна в подпрограмме 2

$d_3$  — константа в подпрограмме 3

## Литература.

1. Киселев В. М., Наильцев В. А. Оценка  
сложности некоторых классов функций.  
Вестник МГУ, 1965, Вып. 4, стр. 44-51.