GENERATION ALGORITHMS OF FAST GENERALIZED HOUGH TRANSFORM

Egor I. Ershov
Moscow Institute of Physics and Technology
141700, 9, Institutskiy per., Dolgoprudny,
Moscow Region, Russia;
Institute for Information
Transmission Problems, RAS
127994, 19, Bolshoy Karetny per.,
Moscow, Russia
E-mail: ershov@iitp.ru

Evgeny A. Shvets
Moscow Institute of Physics and Technology
141700, 9, Institutskiy per., Dolgoprudny,
Moscow Region, Russia;
Institute for Information
Transmission Problems, RAS
127994, 19, Bolshoy Karetny per.,
Moscow, Russia
E-mail: vortexd77@gmail.com

Timur M. Khanipov
Institute for Information
Transmission Problems, RAS
127994, 19, Bolshoy Karetny per.,
Moscow, Russia
E-mail: timur.khanipov@iitp.ru

Dmitry P. Nikolaev
Institute for Information
Transmission Problems, RAS
127994, 19, Bolshoy Karetny per.,
Moscow, Russia
E-mail: dimonstr@iitp.ru

KEYWORDS

Hough transform, fast Hough transform, generalized Hough transform, greedy algorithm, graph matching

ABSTRACT

In this paper we investigate the problem of finding the minimal operations number for the generalized Hough transform computation (GHT). We demonstrate that this problem is equivalent to the addition chain problem and is therefore NP-complete. Three greedy methods for generating GHT computation algorithms are proposed and their performance is compared against the fast Hough transform (FHT) for different discrete straight line pattern types. The additional result of this work is the experimental proof of the FHT non-optimality.

INTRODUCTION

The Hough transform was patented in 1962 by an american scientists Paul Hough for detecting straight tracks in a bubble chamber (Hough. and Arbor 1962) on a photograph taken during an experiment. In 10 years a research was conducted, studying the possibility of using the Hough Transform (HT) for detecting analytical (Duda and Hart 1972) and arbitrary (Merlin and Farber 1975) lines and patterns.

However the suggested algorithms were too demanding on computational power which initiated research about possible reduction in HT computation time and memory resources. In 1972 a principle was suggested to reduce the Hough space dimension for ellipses. This idea used information about possible ellipse center position according to gradient, computed for a given point (Kimme et al. 1975), and can obviously be expanded for any analytical curve.

Later in 1981 D. Ballard published a generalization for all line shapes (patterns) (Ballard 1981) which are computable on a grayscale image, calling it a generalized Hough transform (GHT). After a year in (Davis 1982) an hierarchical GHT computation method for image scale pyramid was suggested which reduced computation time but was still insufficient for real-time systems. In 2003 Ulrich suggested to use hierarchical mapping and R-tables computation (Ulrich et al. 2003), what, as authors say, allows to use this algorithm in real time t create image feature space and solve pattern recognition problems.

An alternative GHT branch was pattern choice randomization during computation (Xu et al. 1990; Xu and Oja 1993). This approach allows to drastically reduce the amount of necessary operations and memory used but does not guarantee an optimal result for a given image. In more detail HT and GHT research and development questions are reviewed in (Illingworth and Kittler 1988; Brown 1992; Mukhopadhyay and Chaudhuri 2015).

Variety of such research prove actuality of fast Hough transform for arbitrary pattern creation problem. Indeed feature extraction is a basic technique in unmanned transport visual algorithms (Konovalenko et al. 2015; Karpenko et al. 2015a,b). Moreover, HT can be useful as an alternative to the methods of pattern recognition (Kuznetsova et al. 2015).

Computing GHT might also be considered as a certain variation of the addition chain problem (Garey and Johnson 1979): for any given pattern (straight line, circle, etc) find such a family of subsets of the pixels set which are to be summed, that the total number of summation operations is minimal. It is worth noting that any binary operation (subtraction, maximum, minimum) might serve as the key

Proceedings 31st European Conference on Modelling and
Simulation ©ECMS Zita Zoltay Paprika, Péter Horák, Kata Váradi,
Péter Tamás Zwierczyk, Ágnes Vidovics-Dancs, János Péter Rádics (Editors)
ISBN: 978-0-9932440-4-9/ ISBN: 978-0-9932440-5-6 (CD)

operation and not only the addition, which substantially expands the GHT computable image feature set. Hence, we have a problem of developing an algorithms generator parametrized by the target pattern and operation types. It should be noted that this approach is in some sense a generalization of the Ulrich's (Ulrich et al. 2003) approach except that the researcher's effort of creating algorithmic constructions accelerating exhaust search should be automated by means of the algorithm generator to be created. We will call the obtained all image-pattern summation algorithm the fast generalized Hough transform (FGHT).

The addition chain problem is NP-complete (Garey and Johnson 1979). In Pippenger's 1980 paper a generalized algorithm for computing the monomials set is suggested (Pippenger 1980), which was further analyzed in more detail in the article (Bernstein ????). Despite this problem's NP-completeness the question of building its approximations with polynomial complexity remains open. For measuring performance of the proposed methods we will compare the constructed algorithm with the fast Hough transform.

PROBLEM STATEMENT

We now formulate the addition chain problem as in (Garey and Johnson 1979). Let C be a family of subsets of a finite set A and J is a positive integer. Is there a sequence

$$S = z_1 \leftarrow x_1 \cup y_1, ..., z_i \leftarrow x_i \cup y_i, i \leq J \qquad (1)$$

of union operators, where each x_i and y_i is either a for some $a \in A$, or z_k for some $k < i$, and for all i, $1 \leq i \leq j$, x_i and y_i do not intersect and for each subset $c \in C$ exists such z_i, $1 \leq i \leq j$, that the corresponding set is equal to c.

We can draw an analogy between the addition chain problem and the problem of constructing the FGHT algorithm. Each element of the A set is an image pixel. A set of patterns for summation (subtraction, maximum or minimum) is given by the C set. In essence we need to find a minimal value J which provides the positive answer for the problem question and produce the corresponding summation sequence.

This problem is NP-complete and further in this paper we show its approximate solution.

GENERALIZED FAST HOUGH TRANSFORM

Consider A elements enumeration with natural numbers, i.e. each c subset can be presented as a vector of natural numbers.

The key idea of our approach is accounting and reusing sums of elements combinations which simultaneously appear in several c subsets. As a replacement we will choose such elements pair p which has the maximal number of occurrences in all patterns n_p (obviously, that is a greedy algorithm). This substitution repeats until each c subset contains one element. Since each such operation reduces the total number of elements in all patterns, this condition has to occur at some point. The case when at a certain step we have several pairs with maximal n_p will be analyzed later.

To implement the algorithm we have to maintain the current n_p values for every pair at each step. For that purpose we will use a hash-table T with increasingly ordered elements with pair p serving as a key and n_p as a value. Assume that elements of each pattern c are also initially ordered by their numbers in increasing order. Before executing the algorithm we fill in T iterating over all patterns c. We will now describe the procedure of fast T value update after each pair substitution. Suppose the new element's e_n number is greater than the maximal one by 1. After substitution we perform pattern content correction during which we consider only the patterns where substitution has taken place. It can be determined by the maximal element value: a substitution occurred if and only if it is equal to the new element's number. For new elements of every pattern where substitution took place, the hash-table is edited: we remove element pairs of kind $\{$pattern element - removed p pair element$\}$ and add a pair $\{$pattern element - new element$\}$. This way of accounting for n_p for all pairs p allows to reduce computation time.

Let us now consider a case when at a certain step we have more than one pair with maximal n_p^*, comprising a set M.

We offer three distinct methods of choosing a substitution pair from M: random choise algorithm, the greedy algorithm and maximum matching search. According to the first one algorithm we randomly choose element of M at each step. We perform computational experiments to investigate properties of this algorithm.

Greedy algorithm

As already mentioned above, the proposed FGHT algorithm is greedy and performs search and substitution for the pair with maximal n_p. Note that maximal n_p cannot change after substitution. We will then choose such $p \in M$ that after substitution we get the biggest number of pairs occurring n_p^* times. In other words, we look for such p which will invalidate minimal number of pairs from M.

To implement such method we should determine how many pairs would become invalidated after p substitution. We make a pass through M using the T table and comparing pairs from M to find equal elements e. If two pairs do not have a common element then it is guaranteed that replacing one of them will not invalidate the other. If they do have a common element, then invalidation will happen. Pseudo-code of this method is depicted in algorithm 1 border.

For the case when several p^* have been found which invalidate equal minimal number of pairs $p \in M$, we construct a set $M^* \in M$ of such pairs. We then again choose such p^* from M^* which invalidate a minimal number of pairs in this set. We repeat the procedure recursively until only one pair is left. If the set size is not reduced during the next iteration, we take a random pair in the result set M^{fin}.

Algorithm 1 Greedy selection algorithm

```
Input: T, M;
Output: p;
n_max = 0;
p_max = 0;
for p ∈ M do
    n = ComputeInvalidateNum(p,M)
    if n > n_max then n_max = n; p_max = p;
    end if
end for
return p;

procedure ComputeInvalidateNum(p,M)
    inv_num = 0;
    for q ∈ M \ p do
        if q and p have equal element number then
            inv_num = inv_num + 1;
        end if
    end for
    return inv_num;
end procedure
```

Graph matching algorithm

We will take the maximal (by number of elements) subset of $p \in M$ not having common pairs and substitute all those pairs at once. Since they do not have common elements, the order of substitutions does not matter.

To use this method one should choose a set of maximal number of pairs which do not have common elements. This problem is similar to finding maximal matching in a graph. Let us build a graph where elements of pairs $p \in M$ are vertices and edges connect elements contained in one of the maximal pairs. Then the maximal edge matching of the graph is exactly the set of edges pairs p of maximal size, such that two distinct pairs do not contain a common element. To find the matching we use the Lemon graph third party library which implements the Blossom matching search algorithm with $O(V^2E)$ complexity (V and E are vertices and edges number resp.).

COMPUTATIONAL EXPERIMENT

To verify the generation algorithm correctness and to estimate the complexity of the generated FGHT we will solve the problem of computing the Hough transform.

In the first experiment we generated FGHT for dyadic patterns (a discrete straight line type used in the fast Hough transform algorithm (FHT) (Nikolaev et al. 2008)) and compare FGHT instructions number with the fast Hough transform method (Götz 1995). In the second experiment we measure complexity of the generated algorithms for Bresenham's lines and compare it with FHT.

In the scope of FHT it is common to consider straight image lines as either primary-horizontal (PH) or primary-vertical (PV), as shown in fig. 1 belonging to one of 4 groups (quadrants):

- PH with right slope $\alpha \in [-\frac{\pi}{4}, 0]$
- PH with left slope $\alpha \in [0, \frac{\pi}{4}]$
- PV with right slope $\alpha \in [\frac{\pi}{4}, \frac{\pi}{2}]$
- PV with left slope $\alpha \in [\frac{\pi}{2}, \frac{3\pi}{4}]$

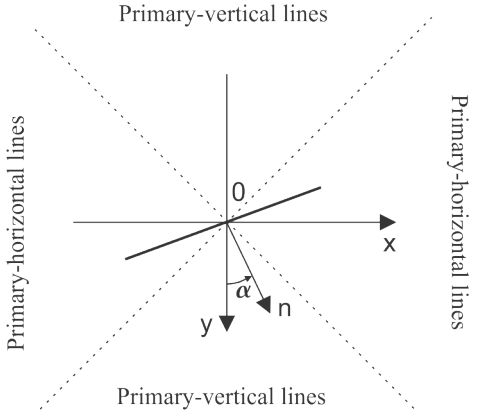

Figure 1: Discrete image straight line types

During computational experiments FGHT generation was performed for all dyadic patterns (Dyadic 4Q), for a half (PH and PV) (Dyadic 2Q) and for a single quadrant (Dyadic 1Q). Apparently, the choice of a particular subgroup for the two latter cases does not matter because of symmetry.

The computational results for three algorithm modifications are given in table 1. Table contains the number of computational operations to compute corresponding Hough transform. One can see that random M element choice version of the FGHT generation algorithm supersedes FHT on operations number per quadrant in all cases. The table shows that greedy M element choice allows to reach a smaller number of operations per quadrant for the Dyadic 2Q and Dyadic 4Q cases, this is explained by appearance of common instructions for different quadrants. The table shows that using the M element choice using maximal matching search does not increase performance even for the Dyadic 2Q and Dyadic 4Q cases.

Random algorithm				
N	FHT 1Q	Dyadic 1Q	Dyadic 2Q	Dyadic 4Q
4	32	36	30	24
8	192	243	223	178
16	1024	1296	1293	1141
32	5120	7330	6773	6399
Greedy algorithm				
N	FHT 1Q	Dyadic 1Q	Dyadic 2Q	Dyadic 4Q
4	32	40	26	24
8	192	208	184	162
16	1024	1024	952	940
32	5120	5120	4784	5083
Maximal matching search algorithm				
N	FHT 1Q	Dyadic 1Q	Dyadic 2Q	Dyadic 4Q
4	32	32	28	23
8	192	244	217	191
16	1024	1457	1048	1151
32	5120	7640	5720	6523

Table 1: Operations number per quadrant for FHT and FGHT generated using random choise, greedy choise and maximal matching search.

During the second experiment FGHT was generated for discrete patterns constructed using the Bresenham

algorithm (Butler et al. 1991).

In fig. 2 it is shown how instructions number depends on square image size N for various modifications. The FHT $N^2 \log N$ complexity is given for comparison for dyadic lines of a single quadrant. FGHT instructions number is given relatively to the number of quadrants.

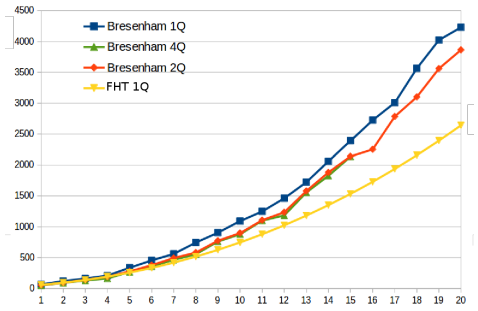

Figure 2: Dependency between instructions number and image size N for FGHT. The Bresenham 1Q curve represents FGHT generated for a single quadrant, Bresenham 2Q - for PH (halved), Bresenham 4Q for all straight lines (divided by 4), FHT 1Q - FHT operations number per single quadrant.

In fig. 2 one can see that FGHT efficiency is slightly less than that of FHT.

It is interesting to notice that computing two quadrants is faster per quadrant than computing just a single quadrant. It means that during two quadrants computation certain common instructions are used, i.e. two quadrants have a greater number of common pairs.

It is not necessarily true, however, that these common instructions are the only reason of greater performance. The gain may also be caused by the fact that when computing two quadrants the first pairs chosen for substitution integrally lead to a more effective algorithm even for a single quadrant. We plan to investigate this question in future works.

The experimental results show that the proposed greedy algorithms do not always demonstrate optimal FGHT performance. This is the case, for example, for images of sizes 4 and 8 when FHT uses less operations.

However, the proposed algorithms do make it possible to generate FGHT which is faster than FHT what apparently means that FHT is not optimal at least for dyadic patterns with $N = 8, 16, 32$.

Table 1 and fig. 2 demonstrate that it is possible to generate FGHT with better performance for dyadic pattern than for the Bresenham algorithm pattern.

We measured generation algorithms execution times on a single CPU core Intel(R) Core(TM) i7-4790 CPU. All three generation algorithms had execution time less than a second for square images with side $N = 4, 8$. However already for $N = 16$ generation time of two algorithms (with randomized and greedy choice) reaches 15 seconds, while the maximal matching algorithm generates FGHT in 1.2 seconds.

For $N = 32$ the randomized, greedy and maximal matching choice algorithms complete computation in 47 minutes, 35 minutes and 4 seconds resp.

These results show that for FGHT generation one may choose between minimizing operations number with greedy choice algorithm showing the best result and the generation time itself where the maximal matching search algorithm wins.

CONCLUSION

In this paper we demonstrated the connection between the FGHT generation and the additive chain problem. We proposed three FGHT generation algorithms for arbitrary patterns and measured their performance in comparison with FHT. It was showed that in certain cases the suggested algorithms outperform FHT by operations number, hence proving that the latter is not always optimal. However, such image sizes exist when FHT has better performance which suggests that the FGHT we constructed is not always optimal either. Based on the obtained results we can state that the FGHT generation approach we suggested has potential and we plan to continue its development.

ACKNOWLEDGEMENTS

We thank Simon Karpenko for advice and inspiration at the beginning of the work. The research was supported by the Russian Science Foundation grant (project No. 14-50-00150).

REFERENCES

Ballard, Dana H. 1981. “Generalizing the hough transform to detect arbitrary shapes.” *Pattern recognition*, 13(2):111–122.

Bernstein, Daniel J. ???? “Pippenger exponentiation algorithm.” *http://cr.yp.to/papers/pippenger.pdf*.

Brown, Lisa Gottesfeld. 1992. “A survey of image registration techniques.” *ACM computing surveys (CSUR)*, 24(4):325–376.

Butler, Nicholas D; Adrian C Gay; and Jack E Bresenham. 1991. “Line generation in a display system.” US Patent 4,996,653.

Davis, Larry S. 1982. “Hierarchical generalized hough transforms and line-segment based generalized hough transforms.” *Pattern Recognition*, 15(4):277–285.

Duda, Richard O and Peter E Hart. 1972. “Use of the hough transformation to detect lines and curves in pictures.” *Communications of the ACM*, 15(1):11–15.

Garey, Michael R and David S Johnson. 1979. “Computers and intractability: a guide to the theory of np-completeness. 1979.” *San Francisco, LA: Freeman*, 58.

Götz, H. J. Druckmüller, W. A. 1995. “A fast digital radon transform—an efficient means for evaluating the hough transform.” *Pattern Recognition*, 28.12:1985–1992.

Hough., Paul V.C. and Ann Arbor. 1962. “Method and means for recognizing complex patterns.”

Illingworth, John and Josef Kittler. 1988. “A survey of the hough transform.” *Computer vision, graphics, and image processing*, 44(1):87–116.

Karpenko, Simon; Ivan Konovalenko; Alexander Miller; Boris Miller; and Dmitry Nikolaev. 2015a. "Uav control on the basis of 3d landmark bearing-only observations." *Sensors*, 15(12):29802–29820.

Karpenko, Simon; Ivan Konovalenko; Alexander Miller; Boris Miller; and Dmitry Nikolaev. 2015b. "Visual navigation of the uavs on the basis of 3d natural landmarks." pages 98751I–987510I.

Kimme, Carolyn; Dana Ballard; and Jack Sklansky. 1975. "Finding circles by an array of accumulators." *Communications of the ACM*, 18(2):120–122.

Konovalenko, I; A Miller; B Miller; and D Nikolaev. 2015. "Uav navigation on the basis of the feature points detection on underlying surface." pages 499–505.

Kuznetsova, E; E Shvets; and D Nikolaev. 2015. "Viola-jones based hybrid framework for real-time object detection in multispectral images." pages 987501N–987506N.

Merlin, Philip M. and David J. Farber. 1975. "A parallel mechanism for detecting curves in pictures." *IEEE Transactions on Computers*, 100(1):96–98.

Mukhopadhyay, Priyanka and Bidyut B Chaudhuri. 2015. "A survey of hough transform." *Pattern Recognition*, 48(3):993–1010.

Nikolaev, D.; S. Karpenko; I. Nikolaev; and P. Nikolayev. 2008. "Hough transform: underestimated tool in the computer vision field." *Proceedings of the 22th European Conference on Modelling and Simulation*, pages 238–246.

Pippenger, Nicholas. 1980. "On the evaluation of powers and monomials." *SIAM Journal on Computing*, 9(2):230–250.

Ulrich, Markus; Carsten Steger; and Albert Baumgartner. 2003. "Real-time object recognition using a modified generalized hough transform." *Pattern Recognition*, 36(11):2557–2570.

Xu, Lei and Erkki Oja. 1993. "Randomized hough transform (rht): basic mechanisms, algorithms, and computational complexities." *CVGIP: Image understanding*, 57(2):131–154.

Xu, Lei; Erkki Oja; and Pekka Kultanen. 1990. "A new curve detection method: randomized hough transform (rht)." *Pattern recognition letters*, 11(5):331–338.

AUTHOR BIOGRAPHIES

EGOR ERSHOV was born in Moscow, Russia. He studied engineer science and mathematics, obtained his Master degree in 2014 from Moscow Institutee of Physics and Technology. Now he is a Ph.D. student. Since 2014 he is working in Vision Systems Lab at the Institute for Information Transmission Problems RAS. His research activities are in the areas of computer vision. His e-mail address is e.i.ershov@gmail.com.

TIMUR KHANIPOV was born in St. Petersburg, Russia. He studied mathematics at the Moscow State University, graduated in 2008. Since 2010 he works as a researcher at the RAS Institute for Information Transmission Problems. Timur's research activities are in the areas of technical vision, industrial automation and recognition systems. His e-mail address is timur.khanipov@iitp.ru.

EVGENY SHVETS was born in Chelyabinsk in 1990. He studied mathematics and obtained his Master degree in 2013 in Moscow Institute for Physics and Technology. Since 2014 he is a research scientist at the Institute for Information Transmission Problems, RAS. His research activities are in the areas of robot localization and mapping and computer vision. His e-mail address is vortexd77@gmail.com

DMITRY NIKOLAEV was born in Moscow, Russia. He studied physics, obtained his Master degree in 2000 and Ph.D. degree in 2004 from Moscow State University. Since 2007 he is a head of the Vision Systems Lab. at the Institute for Information Transmission Problems RAS. His research activities are in the areas of computer vision with primary application to color image understanding. His e-mail address is dimonstr@iitp.ru.