A Gridding Hough Transform for Detecting the Straight Lines in Sports Video

Xinguo Yu
Institute for Infocomm Research,
21 Heng Mui Keng Terrace,
Singapore 119613
xinguo@i2r.a-star.edu.sg

Hoe Chee Lai
School of Computer Engineering,
Nanyang Technological U,
Singapore 639798
s7912516d@ntu.edu.sg

Sophie Xiao Fan Liu
Engineering and Physics Dept.
Oral Roberts University
7777 S. Lewis Ave, Tulsa,
OK 74171, sliu@oru.edu

Hon Wai Leong
Dept. of Computer Science,
National University of Singapore,
Singapore 117543
leonghw@comp.nus.edu.sg

Abstract

*A **gridding Hough transform** (GHT) is proposed to detect the straight lines in sports video, which is much faster and requires much less memory than the previous Hough transforms. The GHT uses the active gridding to replace the random point selection in the random Hough transforms because forming the **linelets** from the actively selected points is easier than from the randomly selected points. Existing straight-line Hough transforms require a lot of resources because they were designed for all kinds of straight lines. Considering the fact that the straight lines interested in sports video are long and sparse, this paper proposes two techniques: the active gridding and linelets process. On account of these two techniques, the proposed GHT is fast and uses little memory. The experimental results show that the proposed GHT is **faster** than the random Hough transform (RHT) and the standard Hough transform (SHT) by 30% and 700% respectively and achieves a 97.5% **recall**, higher than those achieved by either the SHT or the RHT.*

1. Introduction

Detecting straight lines is a key step in many sports video analysis applications because the playfields of many games consist of straight lines. The detected straight lines can facilitate the computer generation of sports video [1-2, 9], the view tracking [7], and player tracking [4]. There have been developed a number of straight-line Hough transforms in the past several decades. However, these transforms are slow in computing and require a large volume of memory because each of them tried to detect all kinds of straight lines. Long computing time especially restricted their applications in the real-time systems because the straight-line detection is one of many steps of these systems; the large memory requirement restricted their applications in cheap devices.

The standard Hough transform (SHT) is expensive in computing because it uses all the edge points to votes. The random Hough tansform (RHT) and the probabilistic Hough transform (PHT) have alleviated this issue of the SHT through randomly selecting subset of all the sample points or all the point pairs to vote [5, 8]. However, two more things were not done yet in their voting procedures. One thing is they did not check whether two sample points are on the same straight line before these two points voted for a cell. Another one is they did not avoid the voting from the isolated points. Hence, they were not robust for some kinds of images. The definition of straight line in mathematics is crystal-clear, but the ones in images processing vary due to different kinds of images in density, straightness, length, and width. Perhaps, *it is better that we develop different algorithms for different kinds of images by using the special characteristics of the considered images if they occur often enough.*

This paper proposes a gridding Hough transform (GHT) for detecting the straight lines in sports video, which is fast and uses little memory. The GHT uses the fact that the interested straight lines in sports video are sparse and long. We do not have interest in short straight lines in sports video because they cannot provide accurate information for further analysis. A long straight line will be cut into several *linelets* (short straight lines) when the given image is gridded into exclusive small blocks and each linelet lies in a block. Furthermore, the two ends of a linelet should be on the boundary of its lying block if the original straight line is continuous. Hence, we work in each block to find linelets. Then we use the linelets from all the blocks to compute a Hough function, measuring how likely a cell corresponds to a real straight line. To avoid using the accumulator, we do statistics on the found linelets. As a result, our measure function uses little memory because it only memorizes the values of a small portion of cells in the Hough space.

The rest of the paper is organized as follows. Section 2 gives the definition of the straight-line Hough transform. Section 3 presents the proposed *gridding Hough transform*. Section 4 shows the experimental results. The conclusion of the paper is given in Section 5.

2. What Is a Straight-Line Hough Transform

In the literature, many straight-line detection algorithms were called Hough, Hough-like, Hough-based algorithms. Almost all the existing Hough algorithms used voting to compute the accumulator, but the voting is not the essential point. As J. Illingworth and J. Kittler [3] has pointed that "The HT converts a difficult global detection problem in image space into a more easily solved local peak detection problem in a parameter space". In other words, the essence of the Hough method is that it converts the shape detection into the problem of analyzing the functions defined on the Hough space. Based on this recognition, we give a general definition of the straight-line Hough algorithm, which has a different view from the definition in [6].

Definition 1 Let F be an image and H be its Hough space for detecting the straight line. Let $M(\bullet)$ be a measure function defined on H. Assume P_1, P_2, and P_3 be three procedures. Then $(M(\bullet); P_1, P_2, P_3)$ is called as a straight-line Hough transform if

1. P_1 can form the straight-line Hough space H of any given image F ;

2. P_2 can compute all the values of $M(\bullet)$ on H.

3. P_3 can find a bunch of cells in H through analyzing $M(\bullet)$, which correspond to the real straight-lines in F.

In this paper, a straight-line Hough transform is also called a (straight-line) Hough algorithm. In the previous Hough algorithms, the result of the voting procedure is a voted accumulator, which can mathematically be viewed as an absolute measure function defined on the Hough space. These measure functions were computed by two kinds of voting procedures. The most Hough algorithms used the **one-to-many** (one point votes for many cells) voting procedure [3, 5]; however, the RHT used **two-to-one (**two points votes for one cell**)** voting procedure [5, 8].

3. Gridding Hough Transform

This section presents the *Gridding Hough Transform* (GHT) for detecting the straight lines in sports video. We start with the overview of the algorithm. Then we present the main techniques: gridding and linelet processing. Last we analyze the complexity and the memory requirement of GHT.

3.1. Outline of the GHT

We observed that the interested straight lines in sports video are long and sparse. Based on this observed fact, this paper proposes a fast Hough algorithm that requires a small volume of memory for detecting straight lines in sports video, which is called the Gridding Hough Transform (GHT). This algorithm uses two new techniques: a gridding method to select points to work on; a linelet processing to acquire the linelets and compute the measure function based on linelets.

Although the random selection in the *random Hough transform* (RHT) and the *probabilistic Hough transform* (PHT) is very simple, it is difficult to find the linelets using them because an arbitrary pair of selected points is probably not from the same straight line. To find linelets easily this paper uses an active point selection method (*gridding method*) to select the edge points. In other words, we grid the given image and the edge point on the grid lines are considered. To improve the quality of the selected points, we shift the start point to grid the given image several times. In the selected edge points by gridding image there are three merits. First, we can locally evaluate whether the pairs on the same grid define linelets. Second, the isolated selected points will not involve the later computing because they are not on any linelet. Thus, the robustness of the algorithm is improved. Last, the linelets formed from the selected edge points are much less than all the points in number. Hence, computing the measure function from the linelets is much faster than from the edge points.

Once we properly grid the given image and identify the edge points on the grid lines, we find the linelets from each block (*grid*). A long straight line should be cut into some linelets. Inversely, these linelets will indicate the existence of the long line. In other words, we determine the existence of the long straight lines from the statistical result of the found linelets. In the mathematical view of point, the statistical result defines a measure function defined on the straight-line Hough space. Each linelet has its line equation, although there is some difference between its equation and the equation of the straight line that this linelet lies on. However, this error is tolerable because the straight lines in an image of sports video are sparse.

3.2. Gridding and Linelet

Our gridding method is to draw horizontal and vertical straight lines on the given image evenly, illustrated in Fig 1. Each square is called a block and four sides of block consist of the boundary of the block. Now we consider the number of edge points on the boundary of a block. A long straight line may have "two points" (which may not two pixels) on the boundary of a block except when it overlays on boundary of the block. Hence, for each block we find the isolated points on its boundary. We ignore the block that has more than τ_1 (a threshold) points on its boundary because such block has a low signal-to-noise ratio. In addition, this ignorance also saves the later computation time.

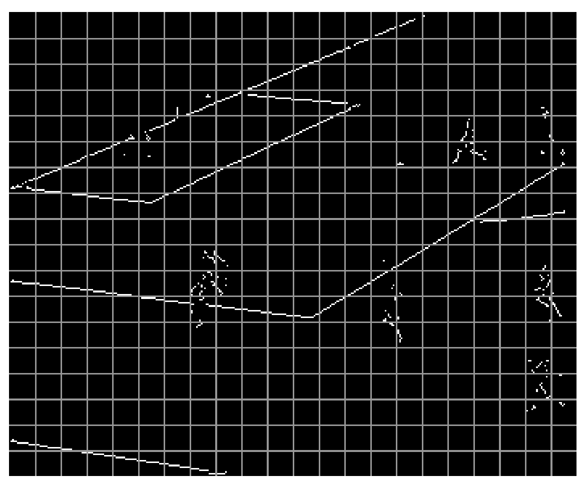
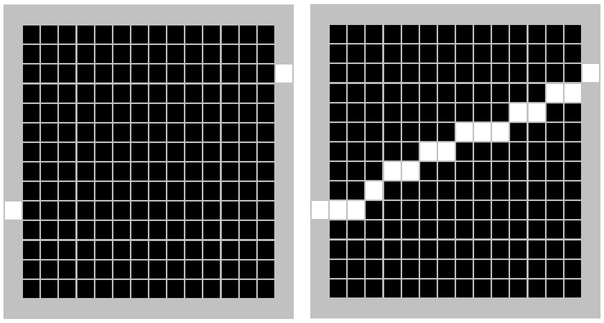


Fig 1. The edge map of a frame with one batch of added grids and found feature points.

Once we identify the edge points on all the blocks, we find the linelets inside each block, illustrated in Fig 2. Each pair of points on the boundary of a block defines a linelet candidate. For each linelet candidate, we count the number of edge pixels that lies on the linelet candidate within the block. This linelet candidate is said to be a linelet if the count is greater than a predefined threshold τ_2 .



(a) (b)

Fig 2. An enlarged gridded block. It is a block in Fig 1 at the 7th row and 8 is the 8th column. (a) shows two points on its boundary; (b) shows all the points between two boundary points.

After we find linelets for all the blocks, we obtain a set of linelets $\Omega = \{K_1, K_2, ..., K_p\}$. Let $K \in \Omega$ be a linelet that has straight line parameters (ρ, θ). Thus, we define the following voting function.

$$V(K, \rho, \theta) = \begin{cases} 1, & \text{if the parameters of K is } (\rho, \theta). \\ 0, & \text{otherwise.} \end{cases} \quad (1)$$

The measure function is defined on the Hough space H based on the voting function as follows.

$$M(\rho, \theta) = \sum_{i=1}^{p} V(K_i, \rho, \theta). \quad (2)$$

Although we define the measure function using the voting function, we compute the measure function using a statistical procedure but not voting procedure as the number of linelets is much smaller than the number of the edge points. Let $\Gamma = \{(\rho, \theta): \text{ a linelet K has straight line parameters } (\rho, \theta)\}$

Procedure 1 (Measure Function Computation)

Step 1: Let $T = \Phi$.

Step 2: For i=1 to p do

Let K_i has a straight line parameter (ρ, θ).

If $(\rho, \theta) \in T$, $M(\rho, \theta) = M(\rho, \theta) + 1$.

Else $T = T \cup \{(\rho, \theta)\}$ and $M(\rho, \theta) = 1$.

The above procedure computes the measure function on the set Γ, which is the final state of T. We have obtained the whole measure function if we define $M(\rho, \theta) = 0$ for all $(\rho, \theta) \in (H - \Gamma)$. It needs a very small volume of memory to store the computed measure function because Γ is much smaller than H. Normally, the cardinal of Γ is less than twice of the number of the straight lines in the given image.

3.3. Gridding Hough Algorithm

Here we give the gridding Hough algorithm, which uses the gridding method and linelet processing developed in the proceeding section.

Algorithm 1

The input is the edge map; the output is the set of detected straight lines.

***Step 1* (*Gridding and Linelet*)**

Grid the map into the small blocks with s being the side length several times.

Find linelets in each block and form the set of linelets $\Omega = \{K_1, K_2, ..., K_p\}$.

***Step 2* (*Computing the Measure Function*)**

Compute the measure function $M(\rho, \theta)$ *through doing statistics on the parameters of all the found linelets, i.e. on* $\Omega = \{K_1, K_2, ..., K_p\}$ *i.e.* ***Procedure 1****.*

***Step 3* (*Find Straight Line Candidates*)**

Consider $(\rho, \theta) \in \Gamma$ *as a straight line candidate if* $M(\rho, \theta) > \tau_2$ *(*τ_2 *is predefined threshold).*

***Step 4* (*Evaluation of Straight Line Candidates*)**

Evaluate whether a straight line candidate is real straight line by checking points that lies around the candidate line.

Tune the parameters of each found straight line.

Find the end points of the found straight line.

3.4. Analysis of Gridding Hough Algorithm

According the definitions in section 2, Algorithm 1 is a Hough algorithm because it converts the straight-line detection into the problem of analyzing the measure function. Now we analyze the time and memory requirement of gridding Hough algorithm.

Theorem 1 The time complexity of the gridding Hough algorithm is $O(w \times h)$, where $w \times h$ is the size of the image.

Proof: The given image is divided into $(w \times h)/s^2$ blocks if the side length of the block is s. For each block, we need 4s operations to find the isolated points on its boundary and $C_{\tau_1}^2 \times s$ operations to verify the linelets. Hence, the total time of the algorithm is $(w \times h)/s^2 \times 7s = O(w \times h)$ when τ_1 is 3.

Theorem 2 The volume of memory that the gridding Hough algorithm requires for storing the measure function is 12q bytes, where q is the number of different (ρ, θ) among all the linelets.

In the real applicatin, q is less than twice of the number of real straight lines in the image. Normally in sports video q is less than 20.

4. Experimental Results

In this paper, the testing images are grabbed from mpeg1 soccer and tennis video files. The soccer video is taken from the match of Senegal-vs-Turkey in the World Cup 2002. We choose 210 frames having various kinds of straight lines from the video. The tennis video is the Men's Final of French Open 2003, which is the game between Juan Carlos Ferrero from Spain and Martin Verkerk from Holland, held on June 8, 2003. We choose 20 frames from the video. For these frames, we manually label the real straight lines to obtain the groundtruth. However, we only label the straight lines that are longer than 60 pixels because our objective is to find long lines. We compare the detected straight lines with the groundtruth to obtain the recall. During the comparison, two conditions must be met before a line classified as a correctly-detected line. The first condition is that the difference between the straight-line parameters must not be greater than the tolerance level. The second condition is that both the endpoints of the detected line must not exceed a predefined number of pixels from the actual endpoints. If either of the endpoints exceeds the predefined number of pixels, then the line equation is classified as false alarm line. False alarm lines are all the detected line candidates by the proposed algorithm but are not classified as found straight lines.

For evaluating our algorithm (GHT), we compare it with the SHT and the RHT. We first compare their computation times. Table 1 shows that our algorithm is faster than the SHT and the RHT by 30% and 720% respectively. In Table 1, Column "time in s" gives the total times in second that three algorithms detect the straight lines in 241 frames; Column "t-p-f in ms" gives the average times in millisecond that three algorithms spend on each frame.

Table 1. Comparison on Running Time.

	time in s	t-p-f in ms	ratio
SHT	11.73	48.672	8.249
RHT	1.912	7.931	1.344
GHT	*1.422*	*5.900*	*1*

Then we compare the recalls of the GHT, the SHT, and the RHT. The GHT has achieved a 97.5% recall. It is much better than the RHT and it is similar to the SHT. In Table 2, the columns "found" and "missing" give the numbers of the found and missing straight lines of the three algorithms.

Table 2. Comparison on Recall.

	found	missing	recall	Groundtruth
SHT	538	19	96.6%	557 straight lines in 241 frames
RHT	445	112	79.9%	
GHT	*543*	*14*	*97.5%*	

The SHT misses some straight lines because the noise points overshade the true straight lines. The GHT alleviates such overshade as the noise points can not form linelets. Hence, the GHT achieves a higher recall than the SHT.

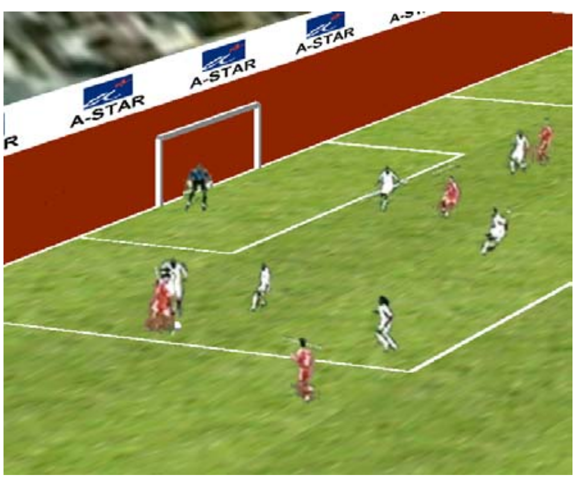


Fig 3. A generated frame of the goalmouth scene using the detected straight lines.

After we detect the straight lines in each frame of sports video, we can generate the goalmouth scene of soccer video using the crossing points of the straight lines as the feature points on the soccer field. Fig 3 shows one of generated frame in such way and you can read paper [9] for more detail on computer generation of video.

5. Conclusions and Future Work

This paper has presented a *gridding Hough transform* (GHT), which is proved to be faster and more robust than the SHT and the RHT for detecting straight lines in sports video. In addition, the presented algorithm uses a very small volume of memory. The contributions of this paper are triple. First, it proposes the gridding method to select subset of points to replace the random selection. Second, it defines the measure function based on linelets. The linelet captures the local property of straight line; the measure function captures the global property of straight line. Thus, the presented algorithm is fast and robust because it uses both local and global properties of straight line. Last, it uses a statistical procedure to get rid of the expensive voting procedure. As a result, the computing time and memory requirement are reduced significantly. This also shows that the voting procedure is not indispensable for a Hough algorithm. In addition, this paper has defined the straight-line Hough algorithm.

In the future we will seek for more methods to obtain linelets besides gridding to detect the straight lines in various images. The algorithm has used to calibrate camera for the goalmouth scene of broadcast soccer video [9]. We will use it to calibrate the camera for tennis video for tennis video reconstruction soon.

6. References

[1] I. D. Reid and A. Zissermann. Goal-directed video metrology, *Proc. of European Conf. on Comp. Vision*, pp647-658, 1996.

[2] D. Farin, S. Krabbe, W. Effelsberg, and P.H.N. de With, Robust camera calibration for sport videos using court models, *Proc. of SPIE, vol. 5307,* 2004.

[3] J. Illingworth and J. Kittler. A survey of the Hough transform, Computer Vision, Graphics, and Image Processing, vol. 44, pp.87-116, 1988.

[4] S. Iwase and H. Saito, Tracking soccer players based on homography among multiple views, (*Visual Communications and Image Processing) Proc. of SPIE*, Vol.5150, pp283-292, 2003.

[5] H. Kälviäinen, P. Hirvonen, L. Xu and O. Erkki. Probabilistic and non-probabilistic Hough transforms: overview and comparisons, *Image and Vision Computing*, vol. 13, pp239-252, 1995.

[6] J. Princen, J. Illingworth, and J. Kittler. A formal definition of the Hough transform: properties and relationships, *J. of Mathematical Imaging and Vision*, vol. 1, 1992, pp153-168.

[7] K. W. Wan, X. Yan, X. Yu, and C. Xu. Real-time goalmouth detection in mpeg soccer video, *Proc. of ACM MM03*, pp. 311-314, 2003.

[8] L. Xu, E. Oja, and P. Kultanen. A new curve detection method: Randomized Hough transform (RHT), *Pattern Recognition Letters*, vol. 11, pp. 331-338, 1990.

[9] X. Yu, Xin Yan, T. S. Hay, H. W. Leong. 3D reconstruction and enrichment of broadcast soccer video, *Proc. of ACM MM04*, pp260-263, 2004.