Implementation of Region-Mura Detection Based on Recursive Polynomial-Surface Fitting Algorithm

WangZhengyao

Fast (shanghai) Imaging Technology
C-101, No.456, Bibo Rd, Pu Dong
Shanghai,China

MaLing

Fast Corporation
2791-5 Shimotsuruma,Yamato
Kanagawa,242-0001,Japan

ABSTRACT

Mura defect appear as low contrast, non-uniform brightness regions in the Flat Panel Display(FPD). Automated Mura inspection method is needed in FPD production. There are many types of algorithms in Mura detection, such as algorithm based on singular value decomposition and surface fitting. Robustness and efficiency is important in the online FPD inspection systems. This paper proposes an efficient Mura detect algorithm for the automatic inspection of FPD. The non-uniform brightness region can be detected by the difference of one fitted Surface and the original image at every pixel. Surface fitting at every pixel is computing cost. This paper proposes a recursive surface fitting method to improve inspection efficiency. Gauss filter is also used in order to segment the non-uniform region. Performance and efficiency of the proposed algorithm has been evaluated on some TFT(Thin FilmTransistor)-LCD panel samples. It is showed that the algorithm is very effective.

KEY WORDS

Mura detection,FPD inspection,polynomial-surface fit

1 Introduction

Quality control is becoming a critical task in the FPD manufacturing process. Surface defects on TFT panel will cause visual failures. Cost and efficient production of FPD demand automatic inspection systems. At present, human visual inspection remains the standard practice for evaluating display luminance and color uniformity through all stages of display development and production. However, human inspection needs higher labor power and usually causes inefficient production and inconsistent inspection: individuals observe and interpret things differently and a single person may report different results over time. Moreover it is difficult for human to inspect large size FPD.

Automated inspection methods use imaging systems combined with scientific digital cameras and sophisticated algorithms. High speed and high resolution digital camera makes automatic inspection possible. Automated defect detection algorithms have also improved to the point that it can replace human visual inspection.

In general, Mura defect appear as low contrast, non-uniform brightness regions. They are larger than a single pixel when the screen is driven to a constant gray level[1]. Depending on the shapes and sizes, Mura defects may be classified into spot-Mura, line-Mura, and region-Mura defect.This paper focus on the region-Mura defects, see Figure1.

Figure 1: Region-Mura in the FPD

Several algorithms have been developed for detection of region-Mura defects[1, 2, 3, 4, 5, 6, 7]. Since the boundary between the regional Mura and the background is indistinct, multiple resolution analysis[2] is used to delete noise and segment the defect region for low-contrast image. For defect detection of periodical, repetitive textured patterns image, Lu and Tsai[3] developed an defect inspection algorithm based on singular value decomposition and reconstruction process. Lee and Yoo[5] use surface fitting and regression to remove the non-uniform background as the first phase to detect candidate region-Mura defects, in the second phase, based on visual perception model, they quantify the candidate Mura defects to identify real muras.

In FPD inspection process, needs of small defects detection on large size FPD demands high speed defect detection algorithms. This paper presents an efficient method for the surface fitting scheme, making it possible for real time FPD automatic inspection systems.

2 Mura Detection based on Surface Fitting

2.1 Basic Defect Detect Algorithm

Lee, Yoo and Choi[5, 6] proposed a surface-fitting based region-Mura detection algorithm. For the sake of performance and efficiency improvement, after some experiments, here we proposed a modified version algorithm. Let $I(r, c), 1 \leq r \leq R, 1 \leq c \leq C$ is the observed

graylevel image of FPD use high resolution digital camera. Then the algorithm works as follows:

I) Input: $I(r,c), 1 \le r \le R, 1 \le c \le C$, window size W, polynomial order d, parameter of Gaussian filter σ, fixed threshold T;

II) Compute the difference of each pixel from the region surround it: for each pixel $p = (r', c')$ in the image I except some marginal pixels;

i) Get square window image centered at p, the square window size is W;

ii) Fit a polynomial-surface model $f^{(d)}(r,c)$ in the square window image use the least-squares method(LSM),

$$f^{(d)}(r,c) = \sum_{i\ge 0, j\ge 0, i+j\le d} \alpha_{i,j} r^i c^j \qquad (1)$$

iii) Compute the absolute value of difference of $dI(r',c') = |I(r',c') - f^{(d)}(r',c')|$

III) Spread the difference of pixel to the region: $D = dI * g(r,c;\sigma)$, where $*$ means linear filtering and $g(r,c;\sigma)$ is Gaussian filter; then D is the region-Mura measures;

IV) Construct a binary image use a single fixed threshold T acquired from experiments.

As described in §2.5, experiments showed that $d = 3$, $W = 101$ and $\sigma = 3.0$ is good for the most images.

There are some key problems in this algorithm: polynomial-surface model $f^{(d)}(r,c)$ fit, Gaussian filtering and single fixed threshold T acquired from experiments. They will be described in the following subsections.

2.2 Two Types Surface Fitting Algorithms

To fit the polynomial-surface model $f^{(d)}(x,y)$ use LSM, the basic algorithm is direct numerical method as described in[8]. Suppose $M = N = 800$ and $W = 101$, there will be $700 \times 700 = 490,000$ times such fits and each fit use $101 * 101 = 10,201$ pixels to compute the model. It is terrible.

The algorithm can be improved by convolution. For the sake of simplicity, here we give a simple example: $W = 3$ and $d = 1$, that is try to fit $f(r,c) = k_1 + k_2 r + k_3 c$ use 9 pixels. Use local coordinate system to describe the pixels in the window,

$$\begin{bmatrix} (-1,-1) & (-1,0) & (-1,1) \\ (0,-1) & (0,0) & (0,1) \\ (1,-1) & (1,0) & (1,1) \end{bmatrix} \qquad (2)$$

corresponding grayvalue are $f_1, f_2, ..., f_9$. Then we have,

$$\begin{bmatrix} f_1 \\ f_2 \\ ... \\ f_9 \end{bmatrix} = \begin{bmatrix} 1 & r_1 & c_1 \\ 1 & r_2 & c_2 \\ ... & & \\ 1 & r_9 & c_9 \end{bmatrix} \begin{bmatrix} k_1 \\ k_2 \\ k_3 \end{bmatrix} \qquad (3)$$

Use Matrix notion, let

$$\mathbf{f} = \begin{bmatrix} f_1 \\ f_2 \\ ... \\ f_9 \end{bmatrix}, \mathbf{A} = \begin{bmatrix} 1 & r_1 & c_1 \\ 1 & r_2 & c_2 \\ ... & & \\ 1 & r_9 & c_9 \end{bmatrix}, \mathbf{k} = \begin{bmatrix} k_1 \\ k_2 \\ k_3 \end{bmatrix} \qquad (4)$$

then

$$\mathbf{f} = \mathbf{A}\mathbf{k} \qquad (5)$$

Obviously, the least-squares solution of $\mathbf{k}$ is

$$\mathbf{k} = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{f} = \mathbf{B}\mathbf{f} \qquad (6)$$

where, $\mathbf{A}_{9\times 3}$ is known, 9 is the number of pixels and 3 is the number of polynomial terms. Then $\mathbf{B}_{3\times 9}$ is also known and can be pre-computed and for each window we can compute k_1, k_2, k_3 using $\mathbf{k} = \mathbf{B}\mathbf{f}$.

Let

$$\mathbf{B} = \begin{bmatrix} b_{11} & b_{12} & b_{13} & \dots & b_{17} & b_{18} & b_{19} \\ b_{21} & b_{22} & b_{23} & \dots & b_{27} & b_{28} & b_{29} \\ b_{31} & b_{32} & b_{33} & \dots & b_{37} & b_{38} & b_{39} \end{bmatrix} \qquad (7)$$

$$\mathbf{b}_1 = [b_{11}, b_{12}, b_{13}, b_{14}, b_{15}, b_{16}, b_{17}, b_{18}, b_{19}] \qquad (8)$$

$$\mathbf{b}_2 = [b_{21}, b_{22}, b_{23}, b_{24}, b_{25}, b_{26}, b_{27}, b_{28}, b_{29}] \qquad (9)$$

$$\mathbf{b}_3 = [b_{31}, b_{32}, b_{33}, b_{34}, b_{35}, b_{36}, b_{37}, b_{38}, b_{39}] \qquad (10)$$

According to (6),we can get

$$k_1 = b_{11}f_1 + b_{12}f_2 + ... + b_{19}f_9 = \mathbf{b}_1 * \mathbf{f} \qquad (11)$$

$$k_2 = b_{21}f_1 + b_{22}f_2 + ... + b_{29}f_9 = \mathbf{b}_2 * \mathbf{f} \qquad (12)$$

$$k_3 = b_{31}f_1 + b_{32}f_2 + ... + b_{39}f_9 = \mathbf{b}_3 * \mathbf{f} \qquad (13)$$

that is, k_1, k_2, k_3 can be computed from linear convolution and the convolution templates, corresponding to each rows of $B_{3\times 9}$, can be pre-computed. This algorithm can be generalized for any W and any d.

So this method transform the polynomial-surface fitting problem to template convolution. Noticed that only k_1 is necessary in computing dI if local coordinate system, like (2), is used. Obviously, the result of this method is equivalent to the direct numerical method but the speed is faster than the direct. But this method also very slow for large W which is needed in region-Mura detection, the computing cost of the method is too high to use.

2.3 Recursive Polynomial-Surface Fitting

In order to get much faster algorithm to fit a polynomial-surface, Haralick, Zuniga and Qiang Ji[9, 10, 11] have proposed many algorithms and the best one is the recursive algorithm.

Def. 1 ***Symmetric Index Set:*** *R is a symmetric index set, if $n \in R$ implies $-n \in R$, let the number of elements in R be N.*

Use $P_n(r)$ express n-order polynomial with the coefficient of the highest ordered item is 1, here call it n ordered polynomial for short. General speaking,

$$P_n(r) = r^n + a_{n-1}r^{n-1} + ... + a_1 r + a_0 \qquad (14)$$

Def. 2 ***Discrete Orthogonal Polynomial on Symmetric Index Set:*** *$\{P_0(r) = 1, P_1(r), ..., P_N(r)\}$ is a group of polynomials, which have $N + 1$ elements, with different degrees, where N is the number of elements of the symmetric index set. It is called discrete orthogonal polynomials on symmetric index set R if*

$$\sum_{r \in R} P_m P_n = 0, 0 \leq m < n \leq N \tag{15}$$

Haralick[9]proposed a recursive formula of discrete orthogonal polynomials on symmetric index set,

$$\begin{aligned} P_0(r) &= 1 \\ P_{i+1}(r) &= rP_i(r) - \beta_i P_{i-1}(r) \end{aligned}$$

where,

$$\beta_i = \frac{\sum_{r \in R} P_i(r) P_{i-1}(r)}{\sum_{r \in R} P_{i-1}^2(r)} \tag{16}$$

So we can get arbitrary degree of discrete orthogonal polynomial on R using this recursive formulas. In our problem, $n = 3$ is enough and

$$\begin{aligned} P_0(r) &= 1 && (17) \\ P_1(r) &= r && (18) \\ P_2(r) &= r^2 - a && (19) \\ P_3(r) &= r^3 - br && (20) \end{aligned}$$

where, $a = \frac{\mu_2}{\mu_0}$, $b = \frac{\mu_4}{\mu 2}, \mu_k = \sum_{r \in R} s^k$.When $R = \{-2, -1, 0, 1, 2\}$, $\mu_0 = 5$, $\mu_2 = 10$, $\mu_3 = 17$, $a = 2, b = 3.4$.

The two dimensional discrete orthogonal polynomial can be constructed from the tensor product of the two sets of one dimensional discrete polynomials. Let $\{P_0(r), P_1(r), ..., P_N(r)\}$ and $\{Q_0(c), Q_1(c), ..., Q_M(c)\}$ be discrete orthogonal polynomials on the symmetric index set$R$ and C respectively. Then $\{P_0(r)Q_0(c), ..., P_m(r)Q_n(c), ..., P_N(r)Q_M(c)\}$ is the discrete orthogonal polynomials on the 2D-symmetric index set $R \times C$.

When $R = C = \{-2, -1, 0, 1, 2\}$, the tensor product discrete orthogonal polynomial on $R \times C$ is $\{1, r, c, r^2 - a, rc, c^2 - a, r^3 - br, (r^2 - a)c, r(c^2 - a), c^3 - br, (r^3 - bc)c, (r^2 - a)(c^2 - a), r(c^3 - bc), (r^3 - br)(c^2 - a), (r^2 - 2)(c^3 - bc), (r^3 - br)(c^3 - bc)\}$. Because we only care polynomials with order not more than $d = 3$, so we should make a selection from above set. Tensor product discrete orthogonal polynomials with degree not exceed $d = 3$ on $R \times C$ are $\{1, r, c, r^2 - a, rc, c^2 - a, r^3 - br, (r^2 - a)c, r(c^2 - a), c^3 - bc\}$, we use $g_i(r, c), i = 1, ..., 10$ to express them.

Obviously, any bivariate cubic polynomial can be expressed by linear combination of those discrete orthogonal polynomials,

$$\begin{aligned} & f(r, c) \\ = \; & k_1 + k_2 r + k_3 c + k_4 r^2 + k_5 rc + k_6 c^2 \\ & + k_7 r^3 + k_8 r^2 c + k_9 rc^2 + k_{10} c^3 && (21) \\ = \; & K_1 + K_2 r + K_3 c + K_4 (r^2 - a) + K_5 rc \\ & + K_6 (c^2 - a) + K_7 (r^3 - b) + K_8 (r^2 - a)c \\ & + K_9 r(c^2 - a) + K_{10}(c^3 - bc) && (22) \\ = \; & \sum_{i=1}^{10} K_i g_i(r, c) && (23) \end{aligned}$$

$K_1, ..., K_{10}$ and $k_1, ..., k_{10}$ have following relations:

$$\begin{aligned} k_1 &= K_1 - aK_4 - aK_6 && (24) \\ k_2 &= K_2 - bK_7 - aK_9 && (25) \\ k_3 &= K_3 - bk_{10} - aK_8 && (26) \\ k_i &= K_i, i = 4, ..., 10 && (27) \end{aligned}$$

Let

$$G(r, c) = \sum_{i=1}^{10} K_i g_i(r, c) \tag{28}$$

so fit the polynomial-surface model $f^{(d)}(x, y)$ use LSM is equivalent to the following problem: minimize e about $K_i, i = 1, ..., 10$,where

$$e = \sum_{(r,c) \in R \times C} [I(r, c) - G(r, c)]^2 \tag{29}$$

Because $g_i(r, c)$ and $g_j(r, c), i \neq j$ is orthogonal,

$$K_i = \frac{\sum_{(r,c) \in R \times C} g_i(r, c) I(r, c)}{\sum_{(r,c) \in R \times C} g_i^2(r, c)} \tag{30}$$

(30) tell us that each coefficient K_i can be computed using $I(r, c)$ by template convolution independently and the corresponding template is W_i,

$$W_i = \frac{g_i(r, c)}{\sum_{(r,c) \in R \times C} g_i^2(r, c)} \tag{31}$$

Compute K_i using W_i is possible but not the best. Because $g_i(r, c)$ is separable,

$$\begin{aligned} g_1(r, c) &= 1 = 1 \times 1 \\ g_2(r, c) &= r = r \times 1 \\ g_3(r, c) &= c = 1 \times c \\ g_4(r, c) &= r^2 - a = [r^2 - a] \times 1 \\ g_5(r, c) &= rc = r \times c \\ g_6(r, c) &= c^2 - a = 1 \times [c^2 - a] \\ g_7(r, c) &= r^3 - br = [r^3 - br] \times 1 \\ g_8(r, c) &= (r^2 - a)c = [r^2 - a] \times c \\ g_9(r, c) &= r(c^2 - a) = r \times [c^2 - a] \\ g_{10}(r, c) &= c^3 - bc = 1 \times [c^3 - bc] \end{aligned}$$

so 2D template convolution can be decomposed into some simple 1D template convolution. For example, template convolution using $g_1(r, c)$ can be decomposed into two 1D template convolution: first template convolution using row template$[1, 1, 1, ..., 1, 1, 1]$ and then using column template $[1, 1, 1, ..., 1, 1, 1]$, here template$[1, 1, 1, ..., 1, 1, 1]$ is generated by polynomial 1.

Now we only focus on some simple 1D template convolution. Suppose the input is $I_1, I_2, ..., I_N$, we are only

interested in following 4 types template convolution:

$$x_n = \sum_{r=-K}^{K} I_{n+r} \quad (32)$$

$$y_n = \sum_{r=-K}^{K} rI_{n+r} \quad (33)$$

$$z_n = \sum_{r=-K}^{K} (r^2 - a)I_{n+r} \quad (34)$$

$$v_n = \sum_{r=-K}^{K} (r^3 - br)I_{n+r} \quad (35)$$

All filtering above use polynomial as filter,

$$\begin{aligned} filter_1(s) &= 1 \\ filter_2(s) &= s \\ filter_3(s) &= s^2 - a \\ filter_4(s) &= s^3 - bs \end{aligned}$$

Now we will construct the recursive formulas of (32), (33), (34), (35).

$$x_{n+1} = \sum_{r=-K}^{K} I_{n+1+r} = x_n + I_{n+K+1} - I_{n-K} \quad (36)$$

$$y_{n+1} = y_n - x_{n+1} + (K+1)I_{n+K+1} + KI_{n-K} \quad (37)$$

$$\begin{aligned} z_{n+1} &= z_n - 2y_{n+1} - x_{n+1} \\ &\quad +[(K+1)^2 - a)]I_{n+K+1} \\ &\quad -[K^2 - a]I_{n-K} \end{aligned} \quad (38)$$

$$\begin{aligned} v_{n+1} &= v_n - 3(z_n - y_n) - (3a - b + 1)x_n \\ &\quad +[K^3 - bK]I_{n+K+1} \\ &\quad +[(K+1)^3 - b(K+1)]I_{n-K} \end{aligned} \quad (39)$$

The formula we proposed here, (39), is different from corresponding formula in Qiang Ji and Haralick's[11]. After carefully examination and numerical experiments, we see that there are some misprints in their paper.

Using (36), (37), (38), (39), we can compute $k_i, i = 1, ..., 10$ of recursively:

I) Input: $I(r, c), 1 \le r \le R, 1 \le c \le C$, $K = (W - 1)/2$;

II) Open float buffer for x, y, z, v, each have the same size of image;

III) Filtering the image row by row:

i) for the Kth pixel in each row, using (32), (33), (34), (35)to compute value of corresponding position in x, y, z, v respectively;

ii) for other pixels in each row, using (36), (37), (38), (39) to compute value of corresponding position in x, y, z, v respectively;

IV) Filtering the image column by column:

i) Filter x using the same method as that in III) to get K_1, K_3, K_6, K_{10};

ii) Filter y using the same method as that in III) to get K_2, K_5, K_6, K_9;

iii) Filter z using the same method as that in III) to get K_4, K_8;

iv) Filter v using the same method as that in III) to get K_7;

v) Compute k_i from K_i using (24), (25), (26) and (27) respectively.

2.4 Gaussian filtering and Mura Region Label

After difference surface fitting image with original image, we can segment Mura defects from the image. Before labeling mura area, Gauss filter can be used to remove the noise.

Gaussian filter is one of the important filters in the image processing and computer vision and then many researchers have studied their properties and implementation. This paper have implemented some of them, and the conclusion is: Deriche'method[12] is a good implementation for parallel computer, Demnigy's method[13] is a good implementation for real-time hardware and Young's method[14] is a good implementation for serial computer. We can choose appropriate method in different system.

The best way to label Mura in the region-Mura measures D is the method based on human visual perception like the ones in Lee, Yoo and Choi[5, 6] and Chen[15]. Even then, this paper try to choose a fixed threshold T, based on human visual perception experiments, for all images.

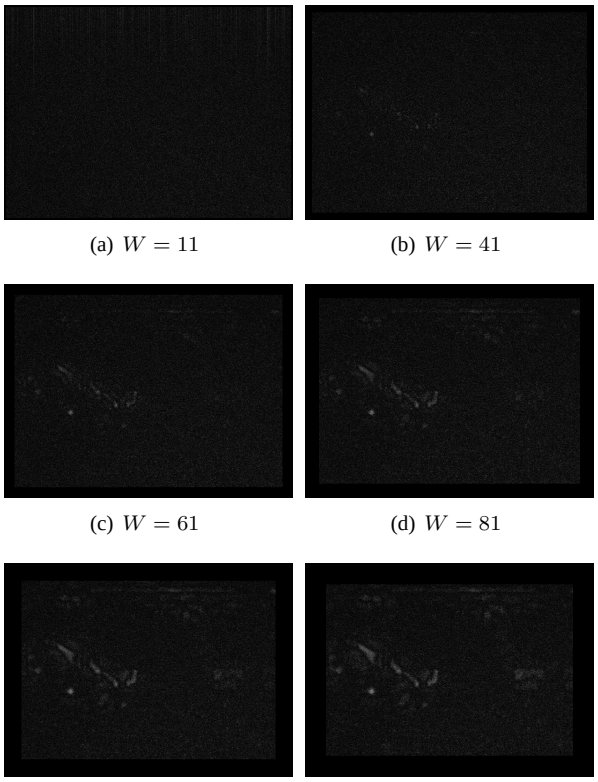
2.5 Experiment and Parameters Selection

Degree of polynomials, d, is important in surface fitting. To get more accurate approximation, higher degree is wanted, but Approximation Theory told that polynomial with degree more than 4 or 5 is improper because of inflexibility. So $d = 3, 4, 5$ is used in most applications. In image processing, $d = 3$ is enough, so this paper choose $d = 3$.

W is the most important parameter in this program. This paper did many experiments, see Figure2, and found that $W = 101$ is the best.

Gaussian smooth is necessary. Figure3 showed that serious texture exists in image and dI is hard to process. In Figure3, (c) is the smoothed image of (a), it is clearly shown that smooth can spread the difference of pixel to the region and eliminate isolated noise which is created by texture. Experiments showed that $\sigma = 3.0$ is best.

The last parameter is T. Some volunteers were asked to observe the images and the results. This paper first use $T = 50.0$ to detect Mura, there is noting in detected image so volunteers dissatisfied the result, then this paper decrease T slowly and detect Mura again. This process was repeated again and again until they satisfied the results. T is different for different volunteers, and the average is $T = 0.35$. So this paper choose $T = 0.35$ as the threshold.

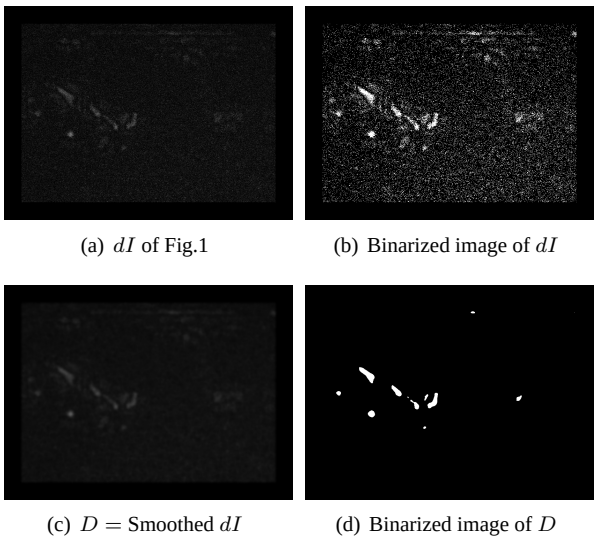


(a) $W = 11$ (b) $W = 41$

(c) $W = 61$ (d) $W = 81$

(e) $W = 101$ (f) $W = 121$

Figure 2: $W = 101$ is good. dI of the image in Figure 1 with various W, $d = 3$. To show the dI clearly, we multiply dI by 10.



(a) dI of Fig.1 (b) Binarized image of dI

(c) $D =$ Smoothed dI (d) Binarized image of D

Figure 3: Smooth is necessary. To show the dI clearly, we multiply dI by 10.

There are some examples of region-Mura detection, with $d = 3$, $W = 101$, $\sigma = 3.0$ and $T = 0.35$, in Figure 4.

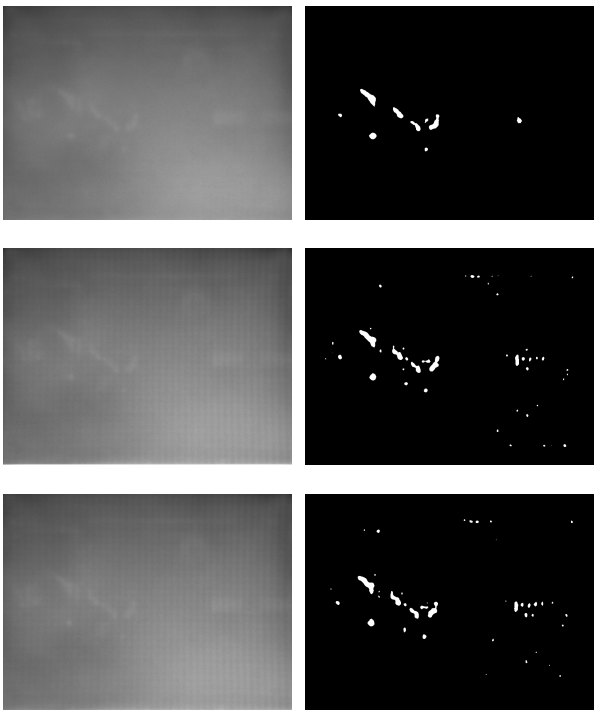


Figure 4: Example of Region-Mura Detection.

Region-Mura may appears in two types of images: with or without periodical, repetitive textured patterns. Experiments showed that this algorithm can be used to detect region-Mura in both of them.

Everyone can download the executable file of the program from my homepage[16].

2.6 Comparison

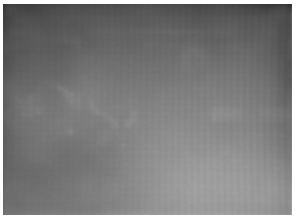
For the same parameters, the performance of recursive method is same as that of template convolution based method, see Figure 5.

Table 1 show that the speed of recursive based implementation is much faster than that of template convolution based.

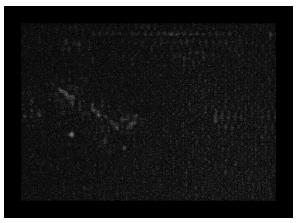
Table 1: Comparison of elapsed time of proposed region-Mura detect algorithm

Implementation	Average Elapsed time (Unit:MSEL)
Template based	12200
Recursive based	65

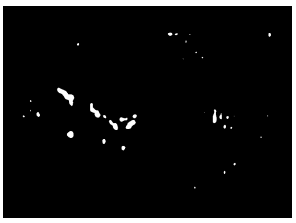
Machine:Intel P4 CPU 3.0GHz;512M Memory
OS:Windows XP sp2;
Program Language of algorithm:Standard C
Compiler: Visual C++ 6.0
Image Size:832×621



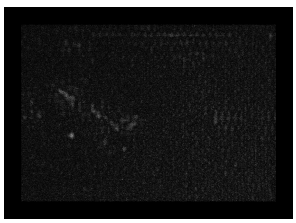
(a) Example image



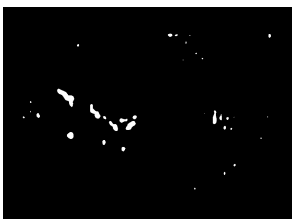
(b) dI based on template based



(c) Final result of (b)



(d) dI based on recursive based



(e) Final result of (d)

Figure 5: Performance comparison between recursive based method and convolution based method

3 Conclusions

This paper proposed a modified version fitting based region-Mura detect algorithm and use recursive polynomial fitting and recursive Gaussian filtering techniques to implement it. The result is good for many images and the program is fast enough in online Mura detection system.

4 Acknowledgments

Thanks to our colleagues, especially LiuWei and JiaBo, who help and encourage us to complete this paper.

References

[1] W.K.Pratt,S.S.Sawkar and K.O'Reilly, "Automatic blemish detection in liquid crystal flat panel displays", IS&T/SPIE Symposium on Electronic Imaging: Science and Technilogy,3306, San Jose,California,January,1998.

[2] Hiroki Nakano, Yumi Mori, "Measurement method for low-contrast nonuniformity in liquid crystal displays by using multi-wavelet analysis", Proc.SPIE Vol. 5880, p. 313-318, Optical Diagnostics; Leonard M. Hanssen, Patrick V. Farrell; Eds 2005.

[3] Chi-Jie Lu and Du-Ming Tsai, "Automatic Defect Inspection for LCDs Using Singular Value Decomposition", Int. J. Adv Manuf Techol(2004).

[4] Chih-Chung Chen, "A Vision System for Dot Defect and Uniformity Automatic Inspection on LCD", Master's Thesis,Mechanical Engineering, 2001.

[5] Jae-Young Lee and Suk-in Yoo,"Automatic detection of region-mura defect in TFT-LCD", IEICE Transactions on Information and Systems, vol.E87-D, no.10, pp.2371-2378, Oct. 2004.

[6] Kyu-Nam Choi and Suk-in Yoo, "Area-Mura detection in TFT-LCD panel", Electronice Imaging, pp.151-158, Jan. 2004.

[7] HC Chen, LT Fang, L. Lee, CH Wen, SY Cheng, and SJ Wang,"LOG Filter Based Inspection of Cluster Mura and Vertical Band Mura on Liquid Crystal Displays", Proceedings of SPIE, Vol. 5679, Jan. 2005.

[8] William H. Press, Saul A. Teukolsky, William T. Vettering, and Brian P Flannery, "Numerical Recipes in C++: The Art of Scientific Computing(Chapter 2)", second edition, Cambridge University Press, 2002.

[9] Robert M.Haralick,"Digital Step Edges from Zero Crossing of Second Directional Derivatives," IEEE Trans. Pattern Analysis and Machine Intelligence, Vol.6, No.1, pp.58-68, JANUARY 1984

[10] O.A. Zuniga and Robert M. Haralick , "Integrated Directional Derivative Gradient Operator," IEEE Trans. Systems, Man, and Cybernetics, vol. 17,no. 3,; pp.508-517,1987.

[11] Qiang Ji and Robert M. Haralick,"Effcient facet edge detection and quantitative performance evaluation",Pattern Recognition 35, pp.689-700.2002.

[12] R. Deriche, Fast Algorithms for Low-Level Vision,IEEE Transactions on PAMI,Vol.12, No.1, pp.:78-87, January, 1990.

[13] Didier Demigny, L. Kessal, J. Pons: Fast Recursive Implementation of the Gaussian Filter. VLSI-SOC 2001: 39-49.

[14] IT Young and LJ van Vliet, "Recursive implementation of the Gaussian filter," Signal Process., vol. 44, pp. 139-151, 1995.

[15] HC Chen, LT Fang, L. Lee, CH Wen, SY Cheng, and SJ Wang, LOG Filter Based Inspection of Cluster Mura and Vertical Band Mura on Liquid Crystal Displays Proceedings of SPIE, Vol. 5679, Jan. 2005.

[16] Download the executable file of the program. http://www.quzhi.net/chinese/vision/mura/rmura.htm.