福山大学工学部紀要
第26巻　2002年11月

カメラキャリブレーションのための ターゲット中心座標の精密計測

藤谷 学*,　大西 芳幸*,　服部 進**

Precise Measurement of Centroid Coordinates of Target Images in Camera Calibration

Manabu Fujitani, Yoshiyuki Ohnishi and Susumu Hattori

ABSTRACT

The precise coordinate measurement of target images is the most crucial factor affecting the performance of camera calibration in vision metrology. Simple binarization techniques to extract target images to pinpoint the center do not work well, when they dot the bright background. In this paper a precise measurement algorithm of target images is discussed by use of the Laplacian of Gaussian (LoG) filter. A window is taken to include each target image is filtered with a LoG filter and its periphery is extracted. The centroid of pixel intensities inside the periphery is calculated. According to an experiment on camera calibration in a two-dimensional target space with bright background, 0.04mm of a posteriori precision of image point coordinates was obtained.

キーワード：画像計測，ディジタルカメラ，キャリブレーション，ターゲット，セントロイド，LoG フィルタ
KeyWords : Vision Metrology, Digital Camera, Calibration, Target, Centroid, LoG filter

1　はじめに

1:50,000 ～ 1:100,000 のオーダの精度を要求する精密工業画像計測では，カメラの内部外部の標定要素と求める対象空間点座標を同時調整しなければならない．しかしそれ以下の計測精度が許容される場合は，オフラインでカメラの内部標定要素を求めておくことがしばしば要求される．

この場合ゴニオメータやコリメータを使った方法よりも，セルフキャリブレーション方式が実用的である．典型的には平坦な壁に反射光を返すレトロターゲットを貼り，さまざまな方向からストロボをたいてこれを撮影する．共線条件から観測方程式を作り，バンドル調整ですべてのパラメータ（内部，外部の標定要素，ターゲットの空間座標）を求める．十分強いネットワークでは内部標定要素は定数になる．

*大学院情報処理工学専攻　　**情報処理工学科

この方法を実用的に用いるにはターゲットの画像座標の中心座標を精密に計測することが大きな課題である．これまでの経験では，十分よい環境でのターゲットの画像座標の計測精度は0.05画素程度である．よい環境とは，ターゲットがよい反射特性と高精度の円形形状を持つこと，壁面が十分暗く反射光を返さないことである．

壁の反射率が低いときには，従来は次の簡単な方法を取っていた．ターゲット像を囲む枠を取りウィンドウを作る．ウィンドウの周辺の画素値 d とその標準偏差 σ を求め，$d+3\sigma$ を閾値として画像を2値化する．これによって中央部のターゲット像が検出される．そのあとターゲット像の領域で画素値を重みとしたセントロイド（重心）の位置を求める．撮影条件がいい場合は，このような単純な方法でも0.05画素の計測精度が出ている[1]．

しかし壁面が明るい場合はこの方法では中心の反射部分だけをうまく抽出できない．ここではラプラシアンガウシアン(LoG)フィルタを使っておよそのエッジを検出する方法を提案し，方法と実験結果を示した[2]．実験によればマニュアルでの計測と比較して約3倍精度が向上し，期待される計測精度0.04画素を得た．

2　ターゲット場と撮影

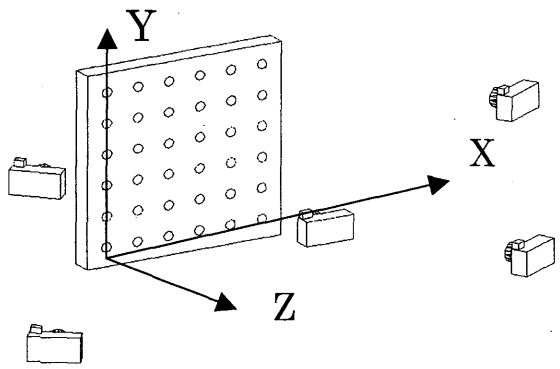


図1　ターゲット，カメラの配置
Fig.1 Configuration of targets and cameras

図1に撮影の様子を示した．直径5mmのレトロターゲット(図2参照)36枚を約200mm間隔に壁に貼った．レトロターゲットの枚数は多いほどいいが，ここでは計測の原理を重視したため，枚数は比較的少ない．このターゲット場をディジタルカメラ，KodakDCS660，（CCDサイズ2K*3K，(1画素10μm)）で撮影した．レンズはNikon20mmである．ターゲット場の中央から1枚，ターゲット場の4隅から中心を狙うように4枚，合計5ヶ所から撮影した．1枚ごとにカメラを光軸周りに90度回転させた．これは光軸周りの標定要素(κ)に変化を与えるためである．絞りを最大の22に絞り，ストロボをたいて撮影し，ターゲットが均質に光るよう，ストロボの強さとCCDの感度を調節した．図3は正面撮影の画像のターゲット画像，および斜めから撮影した画像上カメラから最も遠いターゲット像である．

画像点は全部で180点である．斜めから撮影した画像は反射が弱く暗い．われわれの使っているターゲットは70度までの入射角に対して，60%の光を返す能力がある[3]．

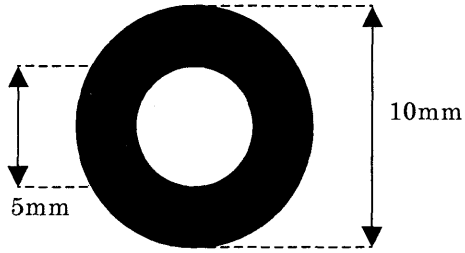
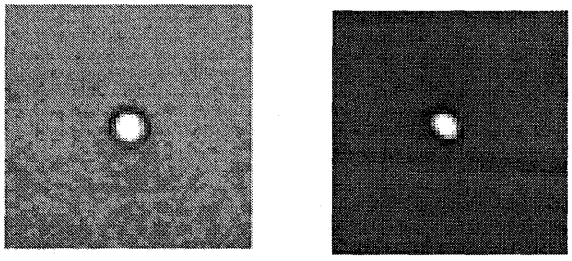


図2　ターゲットの設計値
Fig.2 Design of retro-target



左：正面からの撮影　　右：斜めからの撮影
Left: exposure at front　　Right: exposure at side

図3ターゲットの像
Fig.3 Target images

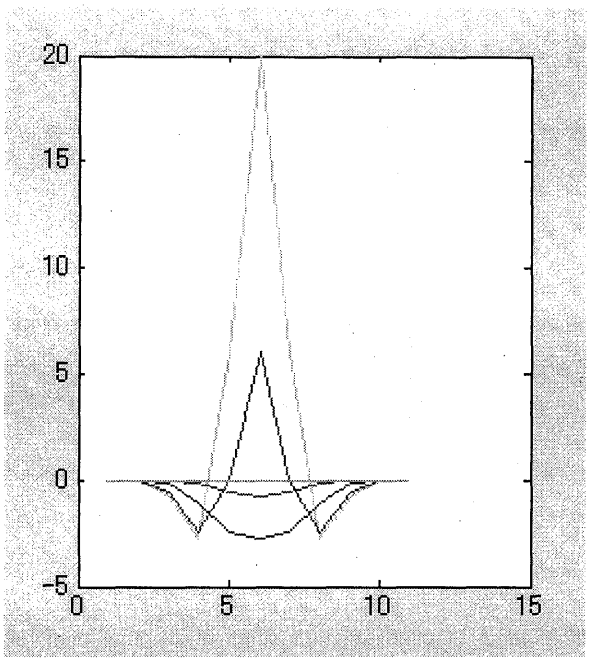


図４　LoG フィルタの断面図
（サイズ 11x11 画素，σ=1.0 ）
Fig. 4 Profiles of a LoG filter.
(size of 11x11 pixels, σ=1.0)

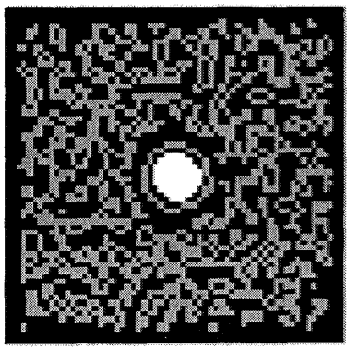


図５　ゼロクロシング(図の灰色)およびターゲット画像の検出(白色)
Fig. 5 Zero-crossings (displayed in gray) and extracted target image (in white).

3　ターゲットの画像座標の計測法

図３のように壁面が明るいとき，ウィンドウ周囲の画素値が大きく，単純な２値化ではうまくターゲット像を取り出せない．またターゲット像が大きいときには周囲のエッジを検出して楕円を当てはめる方法がよく議論される．しかし実際には

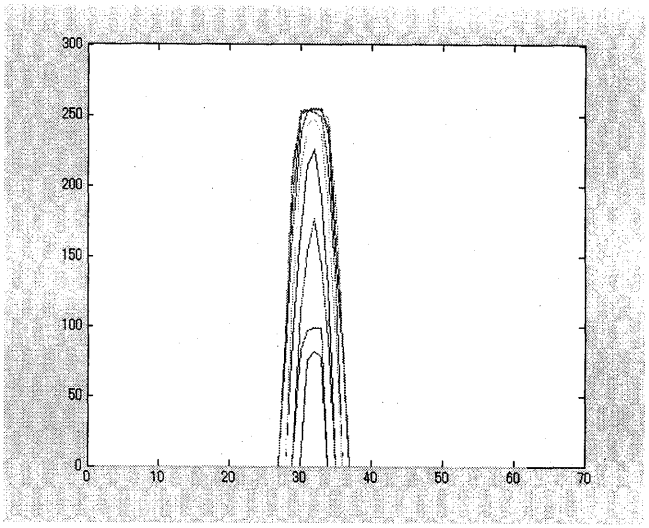


図６　抽出した中心部のターゲット像の画素値分布(行方向に断面図を出力した)
Fig.6 Pixel intensities in an extracted target image (profiles scanned by each row).

ターゲットの像は直径が５画素程度であるので，うまくいかない．ここではつぎのようにターゲット像を抽出し，画素値の重み付き平均を取った．
(1)ターゲットを含むようウィンドウを作る．このため背景の壁の画素値を上回る十分に大きな値を閾値として，画像を２値化してターゲット像のおよその外形を抜出す．ウィンドウのサイズを予想されるターゲット画像の直径の５倍程度に取る．
(2)Laplacian of Gaussian (LoG)フィルタ(式 1) [4]を掛けてそのゼロ交差から画像全体のエッジを検出する．

$$f(x,y)=\left(\frac{\partial^2}{\partial x^2}+\frac{\partial^2}{\partial y^2}\right)\exp\left(-\frac{x^2+y^2}{2\sigma^2}\right) \qquad (1)$$

ここでターゲット像の直径を 10 画素程度とし，LoG フィルタはウィンドウのサイズを 11*11 画素の正方形で実装した．スケールファクタはσ＝1.0 とした．LoG フィルタのサイズはσの 10 倍以上必要である．この値はターゲット像の大きさに応じて変化するべきである．このパラメータではターゲット像が大きい(たとえば直径が 20 画素以上)ときにはターゲット像内にゼロ交差が現れる．図４は LoG フィルタの行ごとの断面図である．

できるだけターゲット像を精度よく取り出すため，ゼロ交差のマイナス側の画素をエッジとした．すなわち１画素余分に外側に取る．図５はウィン

ドウの中でゼロ交差を検出した結果を示した．灰色がゼロクロシングの位置である．
(3)明るさ最大の画素を取り出し，これがターゲット像に入っていると仮定する．これを開始点としdilation 法を使って，(1)で求めたゼロクロシングの画素にぶつかるまで画素を塗りつぶす．つまりターゲット像を囲むゼロ交差の内部をすべて塗りつぶす．図５の中心部の白部分はターゲット像を検出した結果である．図６は抽出した領域での画素値の分布の断面図を示す．
(4)図５の抽出部分に対し，画素値を重みとしてセントロイド（重心）を求め，ターゲットの画像座標とする．

付録１にLoGフィルタリングのプログラム，付録２にセントロイド計算のプログラムを載せた．

4　キャリブレーションの計算法

まず比較のため，手動でターゲット像の中心を計測した．画像を拡大表示しておいて，目測で中心点を指摘しその座標を読んだ．推定計測精度は1/3 画素程度である．一方自動で上記の計測法によって全点の画像座標を計測した．ターゲットが規則的に貼ってあるので，ターゲット画像の同定（レベル付け）は容易である．両者の計測結果に対し，画像座標をBUNDLE調整して，パラメータを求め，画像座標の残差を比べた．座標系XYZは図1に示すように，原点を左下に取り，XY面を壁面上に，Z 軸を手前に来るよう定義した．外部標定要素の近似値と対象点座標の近似値は目分量で入力した．

内部標定要素として，画面距離 c，主点位置 xp, yp，放射方向レンズひずみの係数 $K1, K2, K3$，および接線方向レンズひずみの係数 $P1, P2$ をとった．したがって全部で８個である．レンズひずみの項 $\Delta x, \Delta y$ の式はよく知られている式 2 を採用した[1].

$$\left.\begin{aligned}\Delta x &= -x_p + \left(K_1 r^2 + K_2 r^4 + K_2 r^6\right)\left(x - x_p\right) \\ &+ P_1\left\{r^2 + 2\left(x - x_p\right)^2\right\} + 2P_2\left(x - x\right)\left(y - y_p\right) \\ \Delta y &= -y_p + \left(K_1 r^2 + K_2 r^4 + K_3 r^6\right)\left(y - y_p\right) \\ &+ 2P_1\left(x - x_p\right)\left(y - y_p\right) + P_1\left\{r^2 + 2\left(y - y_p\right)^2\right\}\end{aligned}\right\} \quad (2)$$

基準点がないため，観測方程式には７このランク落ちが生じる．そのため，ターゲットの対象空間座標の平均分散を最小にする拘束を付けて，フリーネット解法で解いた［1］．フリーネットワーク解法では調整計算後に多少座標軸が移動する．

5.キャリブレーション結果

マニュアル観測の時，画像座標の計測精度の事後推定量は σ_0=1.5μm であり，ターゲットの対象空間座標の内的精度(標準偏差)はXYZ方向それぞれで 0.1673mm(X)，0.1698mm(Y)，0.1389mm(Z)であった．

一方LoGフィルタを使った重心計測では，画像座標の事後計測精度は0.47μm であり，マニュアル時より精度が３倍向上した．このときターゲットの対象空間座標の内的精度は 0.0522mm(X)，0.0530mm(Y)，0.0434mm(Z) であった．マニュアルと自動で同じデータを使っているから，対象空間の座標精度はターゲット画像の検出精度で決まり，σ_0 に比例する．

この結果求められた内部標定要素の値は表１のとおりである．ターゲットの数が少ないわりにはいい精度を出しているといえる．

c[mm]	20.31937 ± 4.922e-003
xp[mm]	0.37431 ± 5.132e-003
yp[mm]	-0.04801 ± 5.054e-003
K1[mm^-2]	2.285e-004 ± 6.058e-006
K2[mm^-4]	5.110e-007 ± 1.622e-007
K3[mm^-6]	-6.814e-009 ± 1.304e-009
P1[mm^-1]	-2.389e-007 ± 2.279e-007
P2[mm^-1]	-2.116e-007 ± 2.239e-007

表１　内部標定要素の推定量
Table 1 Estimated interior orientation parameters.

6　結論

２次元ターゲット場を使ったオフラインカメラキャリブレーションで，背後が明るいときにもターゲットの画像座標を精密に求める方法を提案した．

精密キャリブレーションでは，計測ターゲットの像は多いときには5,000を超える．そのため実用にはターゲット像の自動同定と中心座標の自動計測が必至である．前者の問題はターゲットにコ

ードをつけることで解決できる[5].また後者の問題はここで提案する方法を洗練化することで，対処できると考えている．

参考文献

[1] 服部 進, 秋本 圭一, 岡本 厚, 長谷川 博幸, 井本 治孝：ターゲット場の多重撮影による基準点のないCCDカメラキャリブレーション, 電子情報通信学会論文．Vol. J82-D-II, No. 9, pp. 1391-1400, (1998).

[2] 藤谷 学, 大西 芳幸, 服部 進:「カメラキャリブレーションのためのターゲット中心座標の精密計測」, 電気・情報関連学会中国支部第53回連合大会, (2002).

[3] 秋本 圭一：情報化施工のためのデジタル画像計測法に関する研究，京都大学工学博士論文．pp. 76-79, (2002).

[4] 谷口慶治：画像処理工学応用編，共立出版，pp190-192, (2002).

[5] 大西 芳幸, 藤谷 学, 服部 進：コード付きターゲットを用いた全自動カメラキャリブレーション，電気・情報関連学会中国支部第53回連合大会, (2002).

付録1 LoG フィルタリングによってゼロクロシングを計算するプログラム

```
#define size 11
#define gsize 31
int ca[31][31];     //ここに画像の画素値が入っている。

void logfil(void){
    int i,j,k,x[siz],y[siz] ;
    double s,Lf[siz][siz],B[siz][siz],sigma=1.0;
    double cb[gsize+1][gsize+1];
    int cc[gsize+1][gsize+1];

    for(i=0;i<=gsize;i++)
    for(j=0;j<gsize;j++){
        cb[i][j]=0;
        cc[i][j]=0;
    }
    for(i=0;i<siz;i++){
        x[i]=i-5; y[i]=i-5;
    }
    //フィルタかけ～フィルタ作成～
    for(i=0;i<siz;i++)
    for(j=0;j<siz;j++){

    Lf[i][j]=-10*(x[i]*x[i]+y[j]*y[j]-2.0*sigma*sigma)*exp(-(x[i]*x[i]+y[j]*y[j])/(2.0*sigma*sigma));
        //B[i][j]=ca[i][j]*Lf[i][j];
    }
//フィルタかけ
//～実際に画像にフィルタをかける～
    for(i=0;i<gsize-11;i++)
    for(j=0;j<gsize-11;j++){
        s=0;
        for(k=0;k<11;k++)
        for(m=0;m<11;m++)
            s=s+Lf[k][m]*ca[i+k][j+m];

        cb[i+5][j+5]=s;
    }
//フィルタかけ
//～0を交差した場所にチェックを入れる～
    for(i=1;i<gsize;i++)
    for(j=1;j<gsize;j++){
      if(cb[i][j]<=0  &&  (cb[i+1][j]>0  ||
cb[i][j+1]>0 || cb[i-1][j]>0 || cb[i][j-1]>0))
            cc[i][j]=1;
    }
}
    //フィルタかけ～終了～
```

付録2 ターゲット画像のセントロイドを計算するプログラム

```
#define gsize 31
int ca[31][31];
    //対象の画像の画素値が入っている。
int cc[31][31];
    //付録1の結果を使用。ゼロ交差している場所に1が、それ以外の場所には0が入っている。

void cent(int xs,int ys){
    int i,j,sx,sy,ex,ey,ec;
    int ca[gsize+1][gsize+1];
        //対象画像の画素だけ抜き出したもの
    int cc[gsize+1][gsize+1];
```

```
    int maxx=0,maxy=0,max=0;
        //最大値を探すときの変数

//計算対象領域を決める～はじめ、処理開始点の
決定～
//処理開始点を決める。開始点は画素値の一番高
い場所
    for(i=5;i<gsize-5;i++)
    for(j=5;j<gsize-5;j++){
        if(ca[i][j]>max){
            max=ca[i][j]; maxx=i; maxy=j;
        }
    }
    //計算対象領域を決める～領域の決定～
    sx=maxx; ex=maxx; sy=maxy; ey=maxy;
    cc[maxx][maxy]=2;
    ec=0;
    //ec･･･ループ終了かどうかのチェック
  while(ec==0){
    ec=1;
    for(i=sx;i<=ex && i<gsize-5 && i>=5;i++)
    for(j=sy;j<=ey && j<gsize-5 && j>=5;j++){
        if(cc[i][j]==2) {
            ec=0;
            cc[i][j]=3;
            if(cc[i][j+1]==0) {
                cc[i][j+1]=2;
                if(j==ey) ey++;
            }
            else if(cc[i][j+1]==1) cc[i][j+1]=4;
                if(cc[i][j-1]==0) {
                    cc[i][j-1]=2;
                    if(j==sy) sy--;
                }
            else if(cc[i][j-1]==1) cc[i][j-1]=4;
            if(cc[i+1][j]==0) {
                cc[i+1][j]=2;
                if(i==ex) ex++;
            }

            else if(cc[i+1][j]==1) cc[i+1][j]=4;
            if(cc[i-1][j]==0) {
                cc[i-1][j]=2;
                if(i==sx) sx--;
            }
            else if(cc[i-1][j]==1) cc[i-1][j]=4;
        }
    }
    //計算対象領域を決める～終了～

    //重心計算開始
    double gx=0,gy=0,gaso=0;
    int ix=0,iy=0;
    for(i=0;i<gsize-1;i++){
        for(j=0;j<gsize-1;j++){
            if(cc[i][j]==3 || cc[i][j]==4){
                gx+=(double)(ca[i][j]*i);
                gy+=(double)(ca[i][j]*j);
                gaso+=ca[i][j];
            }
        }
    }
    kekkax=gx/gaso;
    kekkay=gy/gaso;

//重心計算終了
    return;

}
```