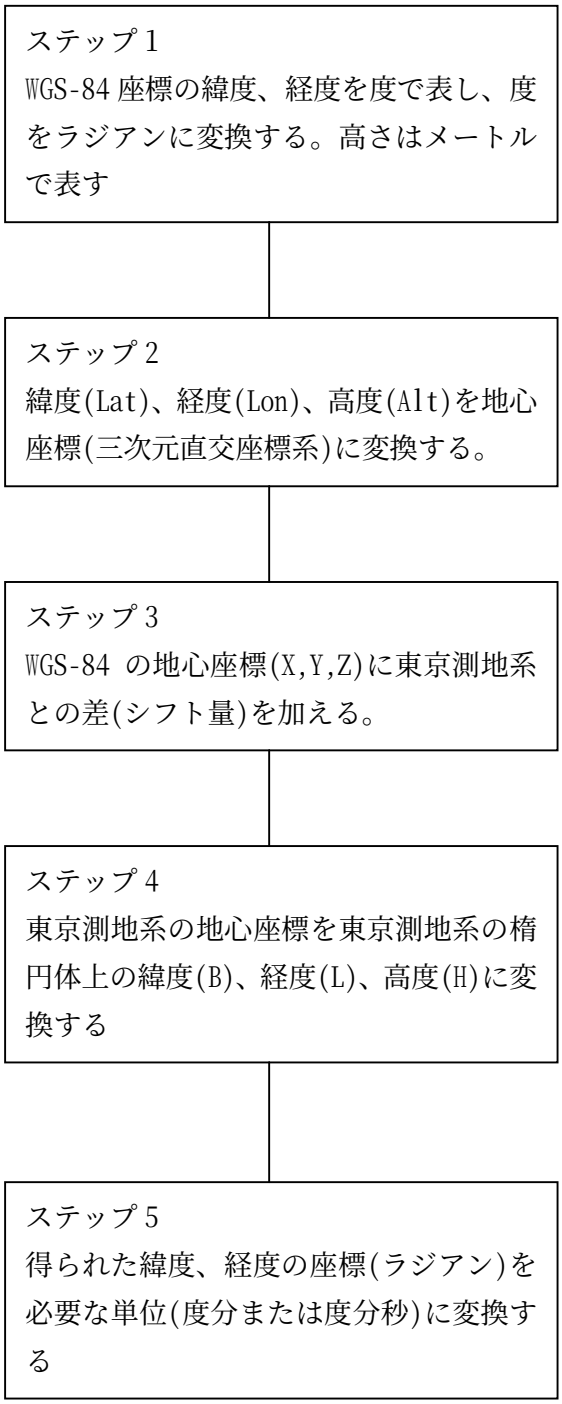
Version 1.01 1998/10/22 S.Yama

座標変換のための資料

1. WGS-84 から東京測地系への変換方法 (3 パラメータシフト法)

1.1 変換のフロー



1. 2. 計算式

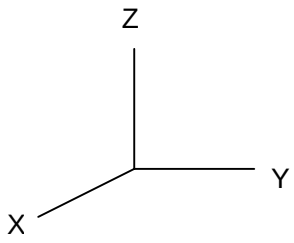
ステップ 1

$$Rad = \frac{Pai}{180} x\, deg$$

ここで、Rad は緯度、経度をラジアンに変換した結果、Pai は円周率(3.141592654)、deg は変換したい緯度、経度を度の単位で表したもの。

ステップ 2

$$X84 = (N + Alt) \times \cos(Lat) \times \cos(Lon)$$
$$Y84 = (N + Alt) \times \cos(Lat) \times \sin(Lon)$$
$$Z84 = (N \times (1 - e^2) + Alt) \times \sin(Lat)$$



ここで、Lat は緯度(ラジアン)、Lon は経度(ラジアン)、Alt は高さ(m)である。
e^2 は以下の通り。

$$e^2 = f84 \times (2 - f84)$$

ここで、f84 = 1/298.257223563

N は以下の式による。

$$N = a / \sqrt{1 - e^2 x \sin^2(Lat)}$$

ここで、a = 6378137.000

ステップ 3

$$X_{Tokyo} = X84 + 147.54$$
$$Y_{Tokyo} = Y84 - 507.26$$
$$Z_{Tokyo} = Z84 - 680.47$$

これらのシフト量の値は、平成 7 年 11 月 30 日付け改正の、建設省公共測量作業規定による。尚、これ以前に公表されているシフト量(dx=146.43、dy=-507.89、dz=-681.46)を使用している場合がある。

ステップ 4
以下をあらかじめ計算しておく。

①

$$b_{Tokyo} = a_{Tokyo} \times (1 - f_{Tokyo})$$

ここで、$a_{Tokyo} = 6377397.155$、$f_{Tokyo} = 1/299.152813$

②

$$P = \sqrt{(X_{Tokyo}^2 + Y_{Tokyo}^2)}$$

③

$$Th = \tan^{-1}\left(\frac{Z_{Tokyo} \times a_{Tokyo}}{P \times b_{Tokyo}}\right)$$

④

$$E^2 = (a_{Tokyo}^2 - b_{Tokyo}^2) / a_{Tokyo}^2$$

⑤

$$Edash^2 = (a_{Tokyo}^2 - b_{Tokyo}^2) / b_{Tokyo}^2$$

緯度(東京測地系)の計算

$$B = \tan^{-1}\left(\frac{Z_{Tokyo} + Edash^2 \times b_{Tokyo} \times \sin^3(Th)}{P - E^2 \times a_{Tokyo} \times \cos^3(Th)}\right)$$

経度(東京測地系)の計算

$$L = \tan^{-1}\left(\frac{Y_{Tokyo}}{X_{Tokyo}}\right)$$

高度(東京測地系)の計算

$$neu = a_{tokyo} / \sqrt{(1 - E^2 \sin^2(B))}$$

$$H = (P / \cos(B)) - neu$$

ステップ 5

$$\mathrm{deg} = \mathrm{Rad} x \frac{180}{\mathrm{Pai}}$$

ここで、deg は変換した結果の緯度、経度を度の単位で表したもの、Rad は得られた緯度、経度で、単位はラジアン、Pai は円周率(3.141592654)。

注意事項:

センチメートルの精度を得るためには必ず倍精度を使用する。
メートルオーダーの精度の場合は、単精度でも可。

Version 0.1 1998/10/22 S.Yama

2．座標系変換プログラムの例

注: 入力と出力の部分は省略されています。

```
/*********************************************************
*      Trnsform.h  for                                   *
*            B,L to X,Y Conversion Program               *
*            X,Y to B,L Conversion Program               *
*            WGS-84 <--> Tokyo datum Vonversion          *
*            For IBM Pc & Compatibles                    *
*                                                        *
*                                  Ver 1.1 2 May 1991    *
*                                      by S,Yama         *
*                                                        *
*********************************************************/

#define a    6377397.155         /* 東京測地系の楕円体の長半径*/
#define a84 6378137.0          /* wgs-84 の楕円体の長半径*/
#define f84 0.0033528106647478     /* 1/298.257223563 WGS-84 の偏平度*/
#define f_tokyo 0.00334277318      /* 1/299.152813 東京測地系の偏平度*/
#define e2 0.006674372231   /*ベッセル楕円体の第１離心率*/
#define e12 0.006719218798   /*ベッセル楕円体の第２離心率*/
#define m0 0.9999   /*縮尺係数 */
#define pi 3.141592653589793   /*円周率*/
#define q 10000855.7658
/* coefficient for equation #1 */
        double A_coef[8]={6366742.52024116306,
-15988.6385238568588,16.7299538817284,-0.0217848007897,0.0000307730631,
-0.0000000453374, 0.0000000000685, -0.0000000000001,};

/* coefficient for equation #2 */
        double B_coef[10] = {-64.7467764, 217.0549893, -283.3714576, 210.1702756,
-156.901395, -637.6283903, 8326.0282307, -39421.8126979,
```

```
 81936.0763069, 9950730.8889188};

/* coefficient for equation #3 */
        double C_coef[9]  = {-0.00000203933504, 0.00000254941046, 0.00001390414031,
-0.00008117092029, 0.000372749204, -0.001796898478,
0.006708984087, -0.01313066154, 1.578708908847};

/* zone origin for latitude */
        double B_origin[17] = {33, 33, 36, 33, 36, 36, 36, 36, 36, 40,
      44, 44, 44, 26, 26, 26, 26};

/* zone origin for longitude */
        double L_origin[17] = {129.5, 131, 132.166666666667, 133.5, 134.333333333333,
136,137.166666666667, 138.5, 139.833333333333, 140.833333333333, 140.25,142.25, 144.25,
142, 127.5, 124, 131};

/*trnsform.h end */

/*******************************************************
*      Jplane.c                                        *
*            B,L to X,Y Conversion Program             *
*            X,Y to B,L Conversion Program             *
*            WGS-84 <--> Tokyo datum Conversion        *
*            For IBM PC & Compatibles,NEC-9801         *
*******************************************************/

/*******************************************************/
/* common functions */
/*度分秒からラジアンへの変換 */
double make_radian(deg, min, sec)
int deg, min;
float sec;
{
        return(((sec/60+min)/60+deg)*pi/180);
}
```

```
/* ラジアンを度分秒へ変換 */
void rad_to_dms(rad,deg,min,sec)
double *rad,*sec;
int *deg, *min;

{
        int d,m,sign;
        double sd;

        sd = fabs(*rad)*3600*180/pi;
        m = floor(sd/60.0);
        sd = sd - (m*60.0);
        d = floor(m/60.0);
        m = m - d * 60.0;
        if (rad < 0) sign = -1;
        else if (rad == 0) sign = 0;
        sign = +1.0;
        d = sign * d;
        *deg = d;
        *min = m;
        *sec = sd;
}
/* common functions end */

/*********************************************************/
/* 東京測地系の緯度、経度を平面座標系に変換 */
/*緯度を与えて赤道からの子午線弧長を求める、出典:精密測地網一次基準点作業規定 P55-56*/

double mx(f)
double f;
{
        double f2,m;
        int i;
        m = A_coef[0]*f;
        for (i = 1; i < 8; i++)
        m = m + A_coef[i]*sin(f*2*i);
```

```
        return(m);
}

/* 計算式の出典:精密測地網一次基準点作業規定 P55-56*/
void bl2xy(double lat,double lon,int zone,double *x,double *y,double *c)
{
        int  cdeg, cmin;
        double f, dl, dx, mx0, csec ;
        double et2,w,n,m,tn2,x2,x4,x6,x8,y1,y3,y5,y7;
        double B0,L0;
        double sinb, cosb, tanb;

        B0 = B_origin[zone] * pi/180;/*原点の緯度*/
        L0 = L_origin[zone] * pi/180;/*原点の経度*/
        dl = lon - L0;
        cosb = cos(lat);
        sinb = sin(lat);
        tanb = tan(lat);
        f = B0;
        mx0 = mx(f);
        f = lat;
        dx = mx(f) - mx0; /*原点の緯度と球点の緯度を与えて、それぞれの赤道からの子午
線からの距離の差を取る。  S-S0*/
        et2 = e12 * pow(cosb,2);
        w = 1 - e2*pow(sinb,2);
        n = a/pow(w,0.5);
        m = n * (1-e2)/w;
        tn2 = pow(tanb,2);
        x2 = n*pow(cosb,2)*tanb*pow(dl,2)/2;
        x4 = n*tanb*pow(cosb,4)*(5-tn2+9*et2+4*pow(et2,2))*pow(dl,4)/24;
        x6              =               n*tanb*pow(cosb,6)*(61-58*tn2+pow(tn2,2)+270*et2-
330*tn2*et2)*pow(dl,6)/720;
        x8              =               n*tanb*pow(cosb,8)*(1385-3111*tn2+543*pow(tn2,2)-
pow(tn2,3))*pow(dl,8)/40320;
        *x = m0*(x8 + x6 + x4 + x2 + dx);
        y1 = n*cosb*dl;
```

```
        y3 = n*pow(cosb,3)*(1-tn2+et2)*pow(dl,3)/6;
        y5 = n*pow(cosb,5)*(5-18*tn2+pow(tn2,2)+14*et2-58*tn2*et2)*pow(dl,5)/120;
/* corrected sign symbols for y5, 1998.10.22 */
        y7 = n*pow(cosb,7)*(61-479*tn2+179*pow(tn2,2)-pow(tn2,3))*pow(dl,7)/5040;
        *y = m0*(y7 + y5 + y3 + y1);
        *c = (sinb*dl+sinb*pow(cosb,2)*(1+3*et2)*pow(dl,3)/3);

}
/* BL to xy conversion end */

/*********************************************************/
/* 平面座標系を東京測地系の緯度、経度に変換 */
/*計算式、係数パラメータは山海堂出版、パーソナルコンピュータによる測量計算プログラムの
第３章による*/
double mxy(double f)
{
        double f2,m;
        int i;

        f2 = pow(f,2);
        m = B_coef[0];
        for (i = 1; i < 10; i++)
                m = m*f2 + B_coef[i];
        return(m*f);
}

double bm(m)
double m;
{
        double m2,b;
        int i;

        m2 = pow(m,2);
        b = C_coef[0];
        for (i = 1;i < 9;i++)
                b = b*m2+C_coef[i];
```

```
        return(b*m);
}

void xy2bl(double XI,double YI,int zone,double *B,double *L,double *c)
{
        double x1,y1,B0,L0,f,m,b1;
        double sinb, cosb, tanb;
        double w, n1, m1, et2, tn2, b2, b4, secb;
        double l1, l3, l5,secl,dl;

        x1 = XI/m0;
        y1 = YI/m0;
        B0 = B_origin[zone] * pi/180;
        L0 = L_origin[zone] * pi/180;
        f = 2*B0/pi;
        m = (mxy(f)+x1)/q;
        b1 = bm(m);
        sinb = sin(b1);
        cosb = cos(b1);
        tanb = tan(b1);
        w = 1-e2*pow(sinb,2);
        n1 = a/pow(w,0.5);
        m1 = n1*(1-e2)/w;
        et2 = e12*cosb;
        tn2 = pow(tanb,2);
        b2 = tanb*pow(y1,2)/(2*m1*n1);
        b4 = tanb*(5+3*tn2+et2-9*et2*tn2-4*pow(et2,2))*pow(y1,4)/(24*m1*pow(n1,3));
        *B = b4-b2+b1;
        l1 = y1/(n1*cosb);
        l3 = (1+2*tn2+et2)*pow(y1,3)/(6*pow(n1,3)*cosb);
        l5 = (5+28*tn2+24*pow(tn2,2))*pow(y1,5)/(120*pow(n1,5)*cosb);
        dl = l5-l3+l1;
        *L = L0+ dl;
        *c = tanb*y1/n1-tanb*(1+tn2-et2)*pow(y1,3)/(3*pow(n1,3))+tanb*(1+tn2)*
        (2+3*tn2)*pow(y1,5)/(15*pow(n1,5));
}
```

```
/* XY to BL conversion end */

/*********************************************************/

/*緯度、経度、楕円体高を三次元直交座標系に変換*/

void dms_to_XYZ(double lat,double lon,double height,
                double *X,double *Y,double *Z,int datum)

{
double e_square,N,f,adms;

        switch(datum)
                {
                case 1: /* wgs-84 */
                adms = a84;
                f = f84;
                break;
                case 2: /* tokyo datum */
                adms = a;
                f = f_tokyo;
                break;
                }
        e_square = f*(2-f);
        N = adms/sqrt(1-e_square*sin(lat)*sin(lat));
        *X = (N+height)*cos(lat)*cos(lon);
        *Y = (N+height)*cos(lat)*sin(lon);
        *Z = (N*(1-e_square)+height)*sin(lat);
}

/*三次元直交座標系から緯度、経度、楕円体高に変換*/
void XYZ_to_dms(double X,double Y,double Z,double *lat,double *lon,
double *height,int datum)
{
double b_dash,P,theta,E_square,E_square_dash,neu,f,adms;
```

```
        switch(datum)
                {
                case 1: /* wgs-84 */
                adms = a84;
                f = f84;
                break;
                case 2: /* tokyo datum */
                adms = a;
                f = f_tokyo;
                break;
                }
        b_dash = adms*(1-f);
        P = sqrt(X*X + Y*Y);
        theta = atan((Z*adms)/(P*b_dash));
        E_square = (adms*adms - b_dash*b_dash)/(adms*adms);
        E_square_dash = (adms*adms - b_dash*b_dash)/(b_dash*b_dash);
        *lat = atan((Z+E_square_dash*b_dash*sin(theta)*sin(theta)*sin(theta))/
                (P-E_square*adms*cos(theta)*cos(theta)*cos(theta)));
        *lon =  atan(Y/X) +pi;
        neu = adms/sqrt(1-E_square*sin(*lat)*sin(*lat));
        *height = P/cos(*lat)-neu;
}

/*3 パラメータ変換、新、旧パラメータの選択に注意*/
void WGS84_to_Tokyo(double *X,double *Y,double *Z)
{
double x,y,z;
        x = *X;
        y = *Y;
        z = *Z;
        *X = x + 146.43;(147.54)
        *Y = y - 507.89;(507.26)
        *Z = z - 681.46;(680.47) ()内は新パラメータ
}

void Tokyo_to_84(double *X,double *Y,double *Z)
```

```
{
double x,y,z;
        x = *X;
        y = *Y;
        z = *Z;
        *X = x - 146.43;(147.54)
        *Y = y + 507.89;(507.26)
        *Z = z + 681.46;(680.47) ()内は新パラメータ
}

/* Datum conversion end */

/*********************************************************/

省略

/* input functions end */

/*********************************************************/
/* Output to the specified file */

省略

/* Write file end */

/*********************************************************/

/* メインプログラムの例、一部省略されている */

main()
{
int mode,input_mode,output_mode;
char more;
int zone,datum;
double lat,lon,height,x,y,c;
double B,L,X,Y,Z;
```

```
double XI,YI,HI;
char *station_id[8] ,*station_name[8];
int i;
```

処理の例

```
        case 1: /* input from keyboard */

        switch(mode)
                {
                case 1: /* BL(Tokyo datum) to XY */
                        read_zone(&zone);
                        read_station();
                        read_BL(&lat,&lon,&height);
                        bl2xy(lat,lon,zone,&x,&y,&c);
                        write_station();
                        write_xy(x,y,c);
                        if ((output_mode == 2) || (output_mode == 3))
                        {
                                fwrite_station();
                                fwrite_xy(x,y,c);
                        }
                        break;
                case 2: /* XY to BL(Tokyo datum) */
                        read_zone(&zone);
                        read_station();
                        read_xy(&XI,&YI);
                        xy2bl(XI,YI,zone,&B,&L,&c);
                        write_station();
                        write_BL(B,L,c);
                        if ((output_mode == 2) || (output_mode == 3))
                        {
                                fwrite_station();
                                fwrite_BL(B,L,c);
                        }
                        break;
                case 3: /*WGS-84 to Tokyo datum */
                        datum = 1;
```

```
                read_station();
                read_BL(&lat,&lon,&height);
                dms_to_XYZ(lat,lon,height,&X,&Y,&Z,datum);
                WGS84_to_Tokyo(&X,&Y,&Z);
                datum = 2;
                XYZ_to_dms(X,Y,Z,&lat,&lon,&height,datum);
                write_station();
                write_BLH(lat,lon,height);
                if ((output_mode == 2) || (output_mode == 3))
                {
                        fwrite_station();
                        fwrite_BLH(lat,lon,height);
                }
                break;
        case 4: /* Tokyo datum to WGS-84 */
                datum = 2;
                read_station();
                read_BL(&lat,&lon,&height);
                dms_to_XYZ(lat,lon,height,&X,&Y,&Z,datum);
                Tokyo_to_84(&X,&Y,&Z);
                datum = 1;
                XYZ_to_dms(X,Y,Z,&lat,&lon,&height,datum);
                write_station();
                write_BLH(lat,lon,height);
                if ((output_mode == 2) || (output_mode == 3))
                {
                        fwrite_station();
                        fwrite_BLH(lat,lon,height);
                }
                break;
        case 5: /* WGS-84 to XYH */
                datum = 1;
                read_zone(&zone);
                read_station();
                read_BL(&lat,&lon,&height);
                dms_to_XYZ(lat,lon,height,&X,&Y,&Z,datum);
```

```
        WGS84_to_Tokyo(&X,&Y,&Z);
        datum = 2;
        XYZ_to_dms(X,Y,Z,&lat,&lon,&height,datum);
        bl2xy(lat,lon,zone,&x,&y,&c);
        write_station();
        write_xy(x,y,c);
        write_height(height);
        if ((output_mode == 2) || (output_mode == 3))
        {
                fwrite_station();
                fwrite_xy(x,y,c);
                fwrite_height(height);
        }
        if (output_mode == 4)
        {
                fwrite_fukui(x,y,c,height);
        }
        if (output_mode == 5)
        {
                fwrite_aisan(x,y,c,height);
        }
        if (output_mode == 6)
        {
                fwrite_jec(x,y,c,height);
        }
        break;
case 6: /* XYH to WGS-84 datum */
        datum = 2;                      /* Tokyo datum */
        read_zone(&zone);
        read_station();
        read_xyh(&XI,&YI,&HI);
        xy2bl(XI,YI,zone,&B,&L,&c);
        dms_to_XYZ(B,L,HI,&X,&Y,&Z,datum);
        Tokyo_to_84(&X,&Y,&Z);
        datum = 1;                      /* WGS-84 */
        XYZ_to_dms(X,Y,Z,&lat,&lon,&height,datum);
```

```
                write_station();
                write_BLH(lat,lon,height);
                if ((output_mode == 2) || (output_mode == 3))
                {
                        fwrite_station();
                        fwrite_BLH(lat,lon,height);
                }
                break;
        }
        break;
```

以下省略