横編地裁断用パターンデータ入力システムの開発　（第３報）

加工技術部　山田　圭、三谷和弘

1. はじめに

本研究は、横編地の裁断で行われているハサミを用いた手裁断を自動化するため、横編地自動裁断装置を開発するものである。今年度はカッターヘッド制御装置へのパターンデータ入力システムについて検討し、その性能を評価するとともに、横編地自動裁断装置の総合調整を行う。

2. 使用機器

(1)　カッターヘッド制御装置

1)　制御装置

2)　制御ソフト作成ツール

3)　インターフェイスユニット

4)　モータ

(2)　直交座標系ロボット

(3)　デジタイザ：グラフテック㈱製

KW-6410

・読 取 方 式　電磁帰還方式

・有効読取範囲　710㎜×470㎜

・分　解　能　0.025㎜以下

3. 内　容

3.1　カッターヘッドの改造

①　これまで試作したカッターヘッドは、オーバーロックミシンのメス機構をそのまま利用していたため下メス把持部の高さが高く、そのため編地を裁断する際には編地を裁断形状に近似した塩ビ板で挟むとともに台座を設け、下メス把持部の高さ分持ち上げる必要があった。しかし、この方式では様々なデザインパターンやサイズ展開、各種

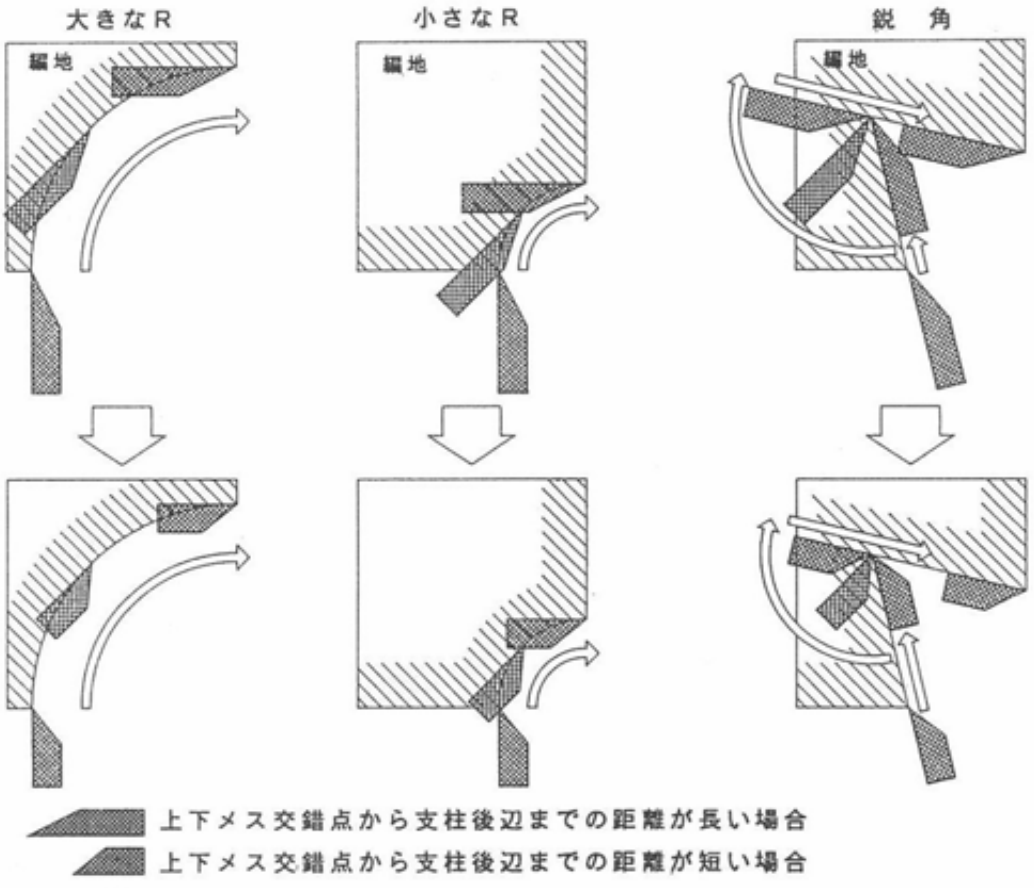


図１　メスの接触範囲

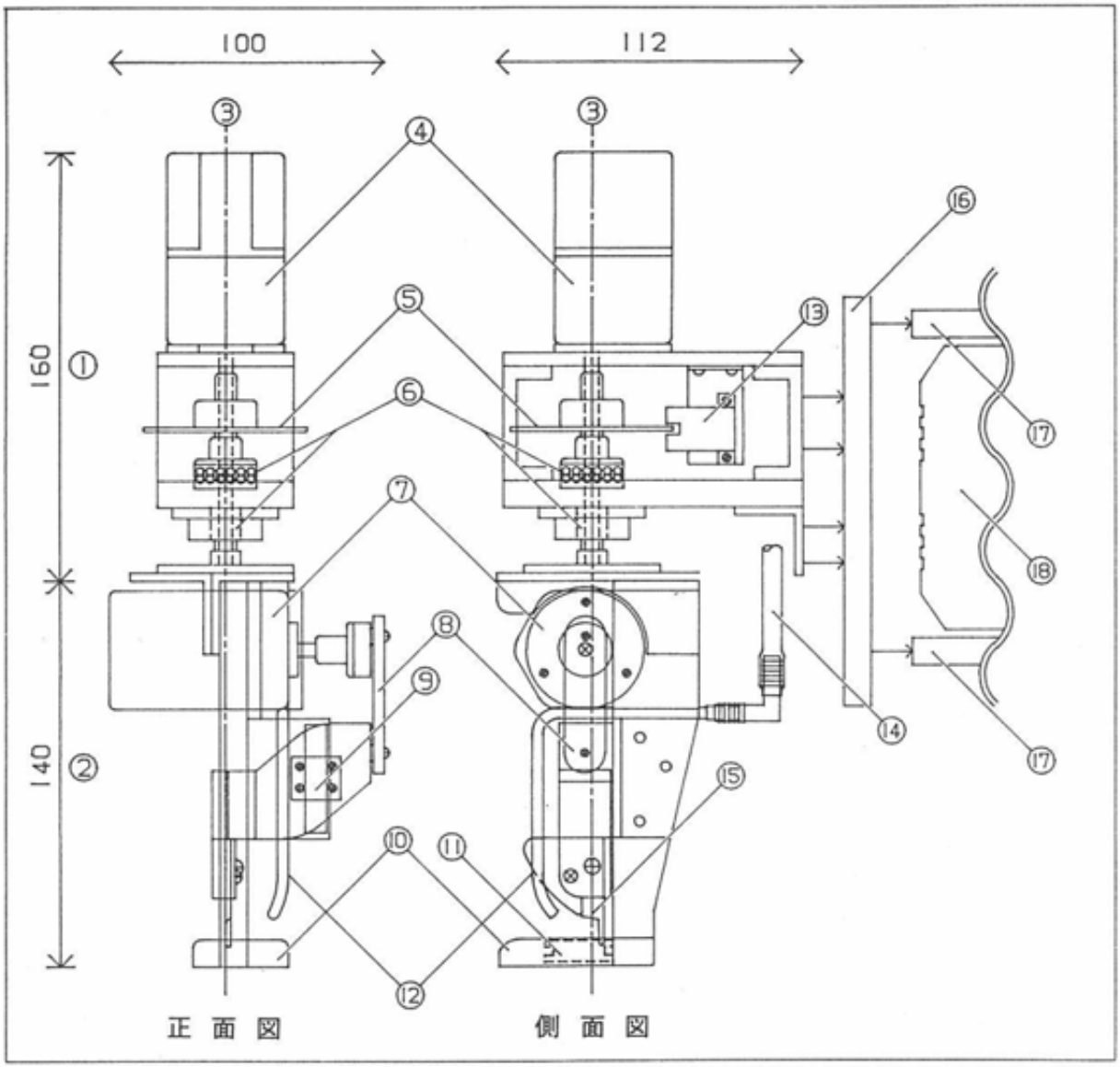
パーツごとに塩ビ板を作成しなくてはならない上、持ち上げた塩ビ板からはみ出して垂れ下がった編地を上下メス交錯点まで誘導するための横編地把持機構を必要とするため構造的に複雑であった。

② カッターヘッドが進行方向に対して右回転を行いながら角、あるいは曲線を裁断する際において、回転角度が大きい場合下メス把持部を支える支柱後辺が編地に接触し、裁断の妨げとなる（図1参照）ことが考えられるため、上下メス交錯点から支柱後辺までの距離は短い方が有利である。

以上①、②を念頭に置き、カッターヘッドの改造を行った。その結果を図2に示す。

図3で示すように、下メスはオーバーロックミシンの下メスを加工し、水平方向に設置することとした。このため、下メス把持部の高さは約10㎜とすることができた。

また上メスは、これまでのカッターヘッドと比較して、より支柱に近接して設置するとともに支柱後辺を研削し、上下メス交錯点から支柱後辺までの距離を短くした。



①Ｒ軸回転機構　②カッターヘッド　③Ｒ軸　④パルスモータ　⑤遮光板　⑥ベアリング
⑦上メス駆動用モータ　⑧クランクロッド　⑨スライド装置　⑩下メス把持部　⑪下メス
⑫エアノズル　⑬フォトセンサ　⑭エアチューブ　⑮上メス　⑯カッターヘッド取付板
⑰直交座標系ロボットスライダ　⑱直交座標系ロボットフレーム（Ｙ軸）

図2　カッターヘッドNo.6及びＲ軸回転機構

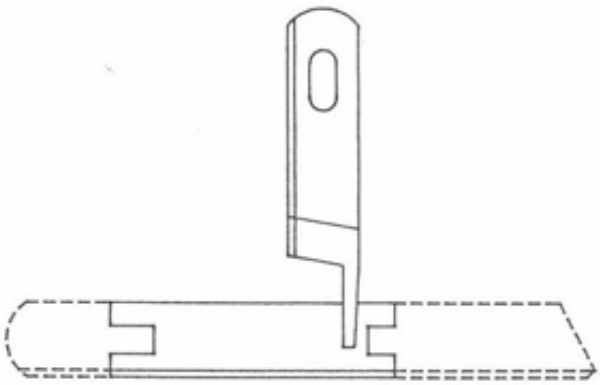


図3　上メス及び下メスの配置

下メス把持部の高さが低くはなったものの、編地を裁断台の上に置いただけでは裁断することはできない。編地は変形しやすい素材であるため、裁断時のわずかな抵抗で歪みが生じ、裁断の妨げとなる。

また、編地は編機によって編み上げられる際編地中心部と左右端に張力差が生じるため、得られた素材は斜行を伴っていることが多い。このため縫製現場では荒く裁断した後セット工程で斜行を矯正するが、それで完全に矯正されない場合もある。この場合、手バサミによる裁断では作業者が斜行を修正しながら切ることができるが、自動で裁断する場合には人手でに斜行を修正した後編地を固定する器具が必要である。この編地固定具として自在定規にホチキスの針を移植したものを試作した。（図4参照）

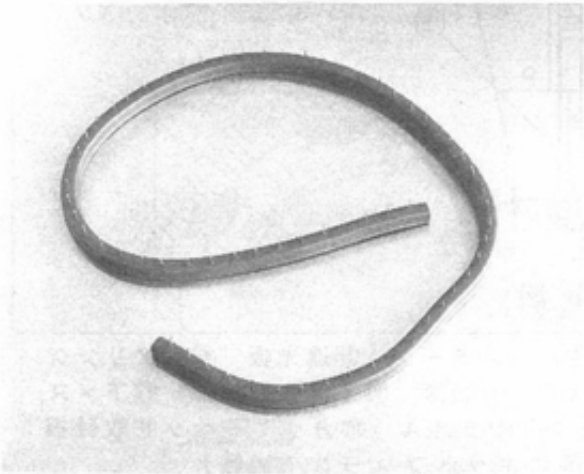


図4　横編地固定具

この横編地固定具は変形可能であるため様々な形状に対応できるが、欠点として、ホチキスの針が危険であること、図5に示すように横編地固定具は裁断線より内側に設置しなくてはならないため、小さなパーツなどは設置する幅が無くなってしまう、ということが上げられる。

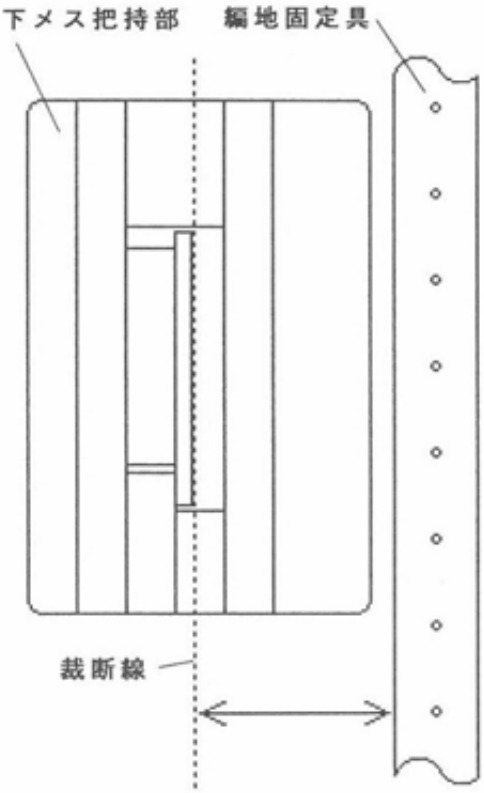


図5　下メス把持部と横編地固定具

型紙形状を入力する装置としてはデジタイザを採用することとし、パターンデータ入力ソフトの開発を行った。

このソフトは、デジタイザからカッターヘッド制御装置にパターンデータをテキストファイルの形式で入力するとともに、そのファイルを元に①カッターヘッド制御装置のディスプレイ画面上へ出力するためのフォーマット、②直交座標系ロボットのX－Y軸制御ソフトへ出力するためのフォーマット、③カッターヘッドのR軸回転制御ソフトへ出力するためのフォーマットに加工するためのものである。図11及び図12にプログラムを、図

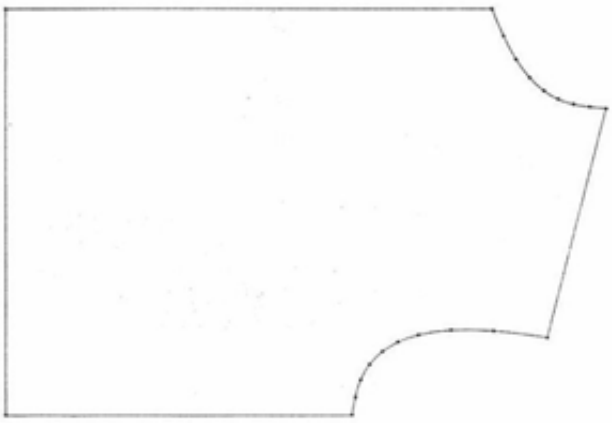


図6　型　　紙

6に入力に使用した前身頃の型紙を、図7にデジタイザからのデータ及びそれを元に加工された各種フォーマットを示す。

デジタイザからのデータは入力されたポイント順に、X座標、Y座標、入力の際に使用されたカーソルのボタンの識別を示す数値が列挙される。

カッターヘッド制御装置のディスプレイ画面上へ出力するためのフォーマットには、1行目にポイントの個数、以降はX座標、Y座標が書き込まれている。また、最後のデータを削除し、最初のデータを複写することにより始点と終点との連結がなされている。

直交座標系ロボットのX－Y軸制御ソフトへ出力するためのフォーマットには、ポイントナンバーと裁断台上でのX座標、Y座標が書き込まれている。Z軸とR軸は使用していないので0が書き込まれている。

カッターヘッドのR軸回転制御ソフトへ出力するためのフォーマットは、1行目にはポイントの個数、以降は回転角度と出力パルス数からなる。

また、カッターヘッド制御装置のディスプレイ画面上へ出力するためのソフトのプログラムを図10に示す。

4. 結果及び考察

4.1　横編地自動裁断装置の総合調整

横編地自動裁断装置の全容を図8に、裁断中のカッターヘッドを図9に示す。

デジタイザによるパターンデータ入力から、裁断作業終了に至るまでの作業の流れを明確化するとともに、各行程における機器の操作及び動作の確認を行った。

パターンデータ入力から裁断作業終了に至るまでの作業の流れを図7に示す。

デジタイザからのパターンデータ	ディスプレイ表示用データフォーマット	直交座標系ロボット制御用データフォーマット	R軸制御用データフォーマット
		#T=MAN3 S=001 E=022	
	0,22		0.000000,22
352 , 353 , 2	35.20,35.30	T1 PT=001 X=+0035.20 Y=+0035.30 Z=+0000.00 R=+0000.00	45.081268,451
2557 , 351 , 2	255.70,35.10	T1 PT=002 X=+0255.70 Y=+0035.10 Z=+0000.00 R=+0000.00	-45.133236,450
2579 , 461 , 4	257.90,46.10	T1 PT=003 X=+0257.90 Y=+0046.10 Z=+0000.00 R=+0000.00	78.742050,787
2611 , 579 , 4	261.10,57.90	T1 PT=004 X=+0261.10 Y=+0057.90 Z=+0000.00 R=+0000.00	-3.863053,38
2671 , 676 , 4	267.10,67.60	T1 PT=005 X=+0267.10 Y=+0067.60 Z=+0000.00 R=+0000.00	-16.566183,165
2751 , 758 , 4	275.10,75.80	T1 PT=006 X=+0275.10 Y=+0075.80 Z=+0000.00 R=+0000.00	-12.553517,125
2850 , 818 , 4	285.00,81.80	T1 PT=007 X=+0285.00 Y=+0081.80 Z=+0000.00 R=+0000.00	-14.488918,144
2980 , 862 , 4	298.00,86.20	T1 PT=008 X=+0298.00 Y=+0086.20 Z=+0000.00 R=+0000.00	-12.519460,124
3191 , 890 , 4	319.10,89.00	T1 PT=009 X=+0319.10 Y=+0089.00 Z=+0000.00 R=+0000.00	-11.139886,110
3461 , 887 , 4	346.10,88.70	T1 PT=010 X=+0346.10 Y=+0088.70 Z=+0000.00 R=+0000.00	-8.195673,81
3808 , 846 , 2	380.80,84.60	T1 PT=011 X=+0380.80 Y=+0084.60 Z=+0000.00 R=+0000.00	-6.101977,60
4182 , 2316 , 2	418.20,231.60	T1 PT=012 X=+0418.20 Y=+0231.60 Z=+0000.00 R=+0000.00	82.464127,825
4077 , 2324 , 4	407.70,232.40	T1 PT=013 X=+0407.70 Y=+0232.40 Z=+0000.00 R=+0000.00	99.917534,999
3977 , 2339 , 4	397.70,233.90	T1 PT=014 X=+0397.70 Y=+0233.90 Z=+0000.00 R=+0000.00	-4.173862,41
3875 , 2371 , 4	387.50,237.10	T1 PT=015 X=+0387.50 Y=+0237.10 Z=+0000.00 R=+0000.00	-8.887252,88
3786 , 2424 , 4	378.60,242.40	T1 PT=016 X=+0378.60 Y=+0242.40 Z=+0000.00 R=+0000.00	-13.355981,133
3693 , 2510 , 4	369.30,251.00	T1 PT=017 X=+0369.30 Y=+0251.00 Z=+0000.00 R=+0000.00	-11.986485,119
3604 , 2626 , 4	360.40,262.60	T1 PT=018 X=+0360.40 Y=+0262.60 Z=+0000.00 R=+0000.00	-9.742638,96
3526 , 2765 , 4	352.60,276.50	T1 PT=019 X=+0352.60 Y=+0276.50 Z=+0000.00 R=+0000.00	-8.197898,81
3451 , 2955 , 2	345.10,295.50	T1 PT=020 X=+0345.10 Y=+0295.50 Z=+0000.00 R=+0000.00	-7.758005,77
353 , 2955 , 2	35.30,295.50	T1 PT=021 X=+0035.30 Y=+0295.50 Z=+0000.00 R=+0000.00	68.459023,685
360 , 358 , 8	35.20,35.30	T1 PT=022 X=+0035.20 Y=+0035.30 Z=+0000.00 R=+0000.00	89.977966,900

図7　デジタイザからのデータ及び各種フォーマット

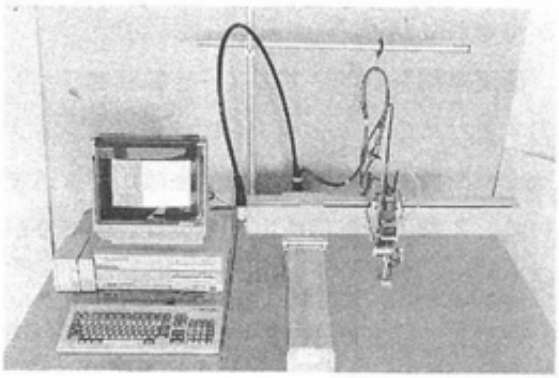


図8　横編地自動裁断装置

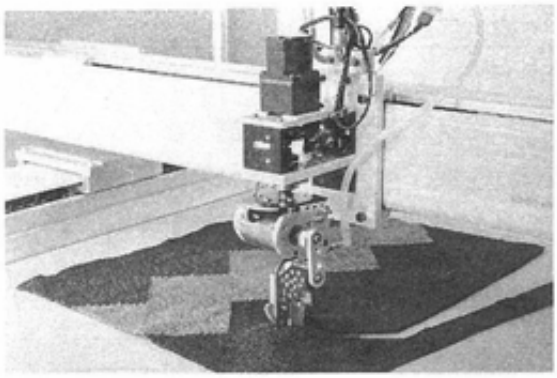


図9　裁断中のカッターヘッド

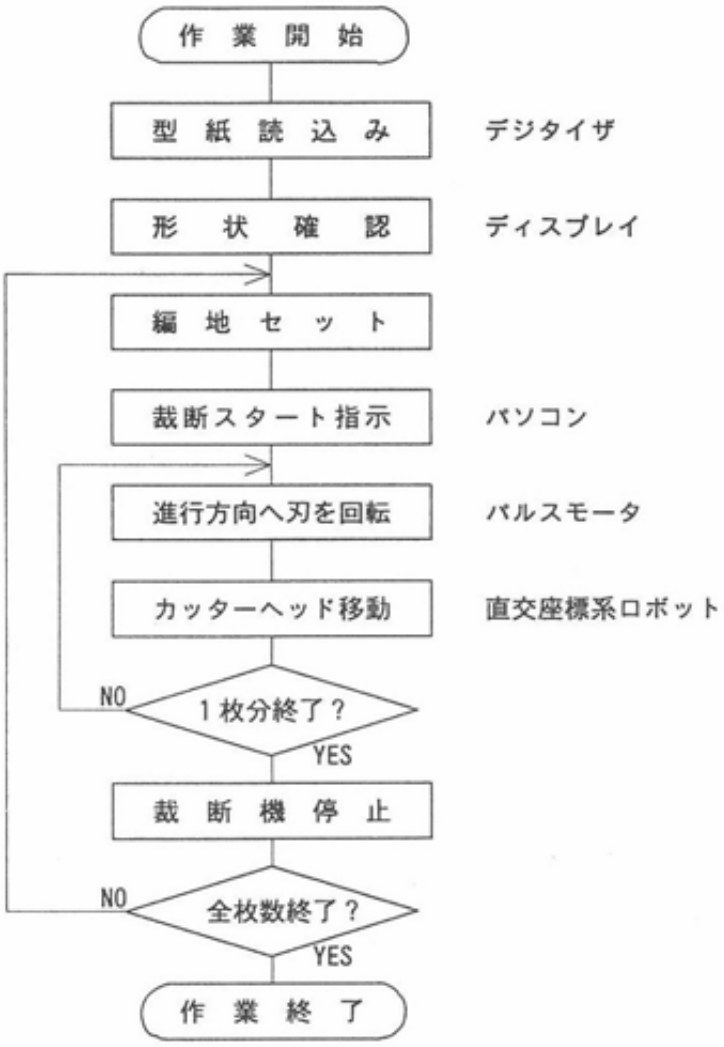


図10　作業の流れ

4.2　実働試験及び性能評価

前身頃の型紙を用い、デジタイザによるパターンデータ入力から裁断作業終了までの実働試験及び性能評価を行った。

先に示した前身頃の周長は約1200㎜であり、これを裁断するのに約60秒を要する。直線におけるカッターヘッドの移動スピードは25㎜／secであり、上メス駆動モータの回転数は1200㎜である

ため上下メスが1回のストロークで裁断する長さは約1.25㎜と算出される。

実働試験の結果、カッターヘッドが方向転換する際、カッターヘッドの支柱後辺が裁断中の編地を圧迫することにより発生する裁断形状の寸法誤差が見られた。また、裁断屑除去のためのエアが強すぎる時にも、編地の変形による裁断形状の寸法誤差が見られた。

5. まとめ

本年度は、横編地裁断の自動化技術に関する研究の最終年度として、パターンデータ入力ソフトの開発を行うとともに、横編地自動裁断装置の総合調整を行った。この結果、手裁断を自動化するための試作機として、基本的には十分な性能を有する装置が開発できた。さらに、この試作機の実用化を考えた場合、以下の問題を解決する必要がある。

① カッターヘッドの小型軽量化

カッターヘッドの下メス把持部を支える支柱をより細くする、下メス把持部自体をより小型化することにより、編地の変形の解消または軽減、回転角度が大きい角または曲線への対応、小さなパーツへの対応等が期待できる。また、軽量化及び重心とR軸ができるだけ一致するようモータ等の配置を考慮し、慣性モーメントを小さくすることによりパルスモータに与える負担も小さくなる。同時に、上メスの駆動による振動も軽減できるため、上メス駆動用モータの回転数を上げることも可能であり、裁断所要時間の短縮にもつながる。

② 裁断所要時間の短縮

デジタイザから型紙形状を入力する際、曲線は直線の連続として捉えている。このため、R軸制御とX－Y軸制御を、パルスモータ、直交座標系ロボットそれぞれの始動、停止信号のやりとりをシェークハンドでコントロールしている現在のソフトでは、曲線(連続した直線)部分の裁断の効率が悪い。これを、曲線にあわせてパルスモータの回転と、直交座標系ロボットの動きを同調させるようなプログラムにすることにより、裁断所要時間が短縮できると考えられる。

③ 裁断された編地の寸法誤差の解消または軽減

裁断された編地の寸法誤差に関しては、前述のように、カッターヘッドの下メス把持部を支える支柱をより細くする、下メス把持部自体わより小型化することによりかなり解消できると思われるが、他の原因として、編地をセットする際の手加減むら、裁断屑除去のためのエアの強弱等も考えられる。

④ 編地固定具の安全性の確保

自在定規を用いた編地固定具は、変形できること、切断して使用することができること等、様々な裁断形状に対応することを考慮すると機能としては悪くはない。しかし、この編地固定具はホチキスの針を用いているため危険であるため、太い針を使用する、針の密度を高める、針の代わりに接着剤を使うなどして安全性を高める必要がある。

⑤ 編地固定具の設置の簡便化

現時点では、下メス把持部の幅の関係で編地固定具は裁断線よりかなり内側に設置しなくてはならず、ペンで下メス把持部の動きをトレースし、それを参考に編地固定具を設置している。

⑥ 各種ソフトの統合および機能拡充

現時点では独立している各ソフトを統合するとともに、型紙入力エラー、裁断エラー等、各種エラーへの対応、入力されたデータの加工、編集、さらには通信によるアパレルCADからの型紙データ取込み等、実際の作業現場での使用に耐えうるソフトにする必要がある。

```
10    CONSOLE 0,25,0,1
20    '
30    CLS 3
40    '
50    COLOR 5,0,0,5
60    PRINT "INPUT FILE NAME(*.TXT)":PRINT
70    '
80    COLOR 6,0,0,6
90    INPUT A$
100   B$=A$+".TXT"
110   '
120   PRINT:PRINT
130   '
140   COLOR 5,0,0,5
150   PRINT"*****************************************"
160   PRINT"*   INPUT PATTERN DATA FROM DIGITIZER  *"
170   PRINT"*****************************************"
180   '
190   PRINT:PRINT:COLOR 7,0,0,7
200   '
210   OPEN "COM:N73" FOR INPUT AS #1
220   OPEN B$ FOR OUTPUT AS #2
230   PRINT #1,"P";
240   INPUT #1,X,Y,F
250   PRINT X:Y:F
260   PRINT #2,X:",":Y:",":F
270   IF F<>8 THEN 240
280   CLOSE #1
290   CLOSE #2
300   '
310   PRINT:PRINT:COLOR 4,0,0,4
320   PRINT "!!"B$;" WAS CREATED !!"
330   '
340   COLOR 5,0,0,5
350   '
360   PRINT:PRINT:PRINT
370   PRINT "IS THIS DATA OK ? (YES-Y,NO-N)"
380   INPUT C$
390   IF C$="Y" THEN 430 ELSE 410
400   '
410   KILL B$:GOTO 30
420   '
430   COLOR 7,0,0,7
440   END
```

図11　パターンデータ入力のためのプログラム

```
/*************************************************************
*    WKDAT.C            デジタイザからの生データを加工          *
*  (*.txt→*.dtz)(*.dtz→*.tp3)(*.dtz→*.sq3)(*.dtz→*.pmd)    *
*************************************************************/
#include        <stdio.h>
#include        <string.h>
#include        <math.h>
#include        <stdlib.h>
int ii,iii,ff;
int pntno;/* 座標データ個数 */
int pntcnt=1;/* 座標指示カウンタ */
int pulsesuu;
float xx,yy,zz,xx1=0,yy1=0,xx2,yy2,ang,ang1=0,ang2;
char name[80];
char filename[80];
const char *d_drive = "d:";
char txtname[80];
char dtzname[80];
char tpname[80];
char sqname[80];
char pmdname[80];
void txtread(char *);
void dtzwrite(char *);
void dtzread(char *);
void tpwrite(char *);
void sqwrite(char *);
void pmdwrite(char *);
struct txtpoint
    {    float x;
         float y;
         int   f;
    }    txtpnt[512];
struct dtzpoint
    {    float x;
         float y;
    }    dtzpnt[512];
struct pmdpoint
    {    float m;
         float n;
    }    pmdpnt[512];
int main()
{
    printf("¥33[2J");/* テキスト画面のクリア */
    strcpy(filename,d_drive);
    printf("¥33[33m");/* テキストカラーの設定 */
    printf("INPUT FILE NAME(*.txt)¥n");/* ファイル名の入力 */
    printf("¥33[m");/* テキストカラーを元に戻す */
    scanf("%s",name);
    strcat(filename,name);
    strcpy(txtname,filename);
    strcat(txtname,".txt");
    txtread(txtname);
    printf("¥33[32m");
    printf("¥nTOTAL %d POINTS¥n",ii);/* データ個数の読み取り */
    printf("¥n");
    for (iii=0;iii<=ii-1;iii++) /* 座標読み取りデータ読み出し */
        {
        xx=txtpnt[iii].x;
        yy=txtpnt[iii].y;
        ff=txtpnt[iii].f;
        printf("P%2d(%6.2f ,%6.2f),%d¥n",iii+1,xx/10,yy/10,ff);
        }
    printf("¥33[m");
    strcpy(dtzname,filename);
    strcat(dtzname,".dtz");
    dtzwrite(dtzname);
    strcpy(dtzname,filename);
    strcat(dtzname,".dtz");
    dtzread(dtzname);
    strcpy(tpname,filename);
    strcat(tpname,".tp3");
    tpwrite(tpname);
    strcpy(sqname,filename);
    strcat(sqname,".sq3");
    sqwrite(sqname);
    strcpy(pmdname,filename);
    strcat(pmdname,".pmd");
    pmdwrite(pmdname);
    return 0;
}
/*************************************************************
*              .txtファイルから読み出し、配列に格納          *
*************************************************************/
void txtread(char *txtname)
{
    FILE *fp;
    float fpx,fpy;
    int i,fpf;
    fp = fopen(txtname,"r");
    if (fp == NULL)
    {
        printf("¥33[41m");
        printf("¥nCANNOT OPEN %s!!¥n",txtname);
        printf("¥33[m");
        exit(1);
    }
    else
    {
        i = 0;
        while (fscanf(fp," %f , %f , %d ",&fpx,&fpy,&fpf) !=EOF)
        {     /*配列へ入れる*/
            txtpnt[i].x=fpx;
            txtpnt[i].y=fpy;
            txtpnt[i].f=fpf;
            i++; ii=i;
        }   fclose(fp);
    }
}
```

図12－1　デジタイザからのデータを各種フォーマットに加工するためのプログラム

```
/*************************************************************
*            始点と終点の結合及び*.dtzファイルへの書き込み    *
*************************************************************/
void dtzwrite(char* dtzname)
{
    FILE *fpo;
    float fpx,fpy;
    int fpf;
    int i;
    fpo = fopen(dtzname,"w");
    if (fpo == NULL)
    {
        printf("¥33[41m");
        printf("¥nCANNOT OPEN %s!!¥n",dtzname);
        printf("¥33[m");
        exit(1);
    }
    else
    {
        fprintf(fpo,"0,%d¥n",iii);/* *.dtzの最初の行(ﾎﾟｲﾝﾄ数)*/
        printf("¥n");printf("¥n");
        printf("TOTAL %d POINTS¥n",iii);/*画面表示*/
        printf("¥n");
        for (i=0;i<=iii-2;i++)/*最後のデータは無視*/
        {    /* 配列から読み、 ファイルへ書き込む*/
            fpx=txtpnt[i].x;
            fpy=txtpnt[i].y;
            fpf=txtpnt[i].f;
            fprintf(fpo,"%.2f,%.2f¥n",fpx/10,fpy/10);
            printf("P%2d(%6.2f ,%6.2f)¥n",i+1,fpx/10,fpy/10);
        }
/*最後のデータ（終点）に始点のデータを入れる*/
        fprintf(fpo,"%.2f,%.2f¥n",
                txtpnt[0].x/10,txtpnt[0].y/10);
        printf("P%d(%6.2f ,%6.2f)¥n",iii,
                txtpnt[0].x/10,txtpnt[0].y/10);/*画面表示*/
        fclose(fpo);
    printf("¥33[32m");
    printf("¥nLAST POINT WAS JOINTED TO FIRST POINT¥n");
    printf("¥33[m");
    }
    printf("¥33[36m");
    printf("¥n%s WAS CREATED !!¥n",dtzname);
    printf("¥33[m");printf("¥n");printf("¥n");
}
/*************************************************************
*              .dtzファイルから読み出し、配列に格納          *
*************************************************************/
void dtzread(char *dtzname)
{
    FILE *fp;
    float fpx,fpy;
    int i;
    fp = fopen(dtzname,"r");
    if (fp == NULL)
    {
        printf("¥33[41m");
        printf("¥nCANNOT OPEN %s!!¥n",dtzname);
        printf("¥33[m");
        exit(1);
    }
    else
    {
        i = 0;
        while (fscanf(fp,"%f,%f",&fpx,&fpy) !=EOF)
        {    /*配列へ入れる*/
            dtzpnt[i].x=fpx;
            dtzpnt[i].y=fpy;
            i++; ii=i;
        }   fclose(fp);
    }
}
/*************************************************************
*                          *.tp3ファイルへの書き込み          *
*************************************************************/
void tpwrite(char* tpname)
{
    FILE *fpo;
    float fpx,fpy;
    int i;
    fpo = fopen(tpname,"w");
    if (fpo == NULL)
    {
        printf("¥33[41m");
        printf("¥nCANNOT OPEN %s!!¥n",tpname);
        printf("¥33[m");
        exit(1);
    }
    else
    {
      fprintf(fpo,"#T-MAN3 S-001 E-%03d           ",iii);
      fprintf(fpo,"                                ");
      fprintf(fpo,"                               ¥n");
      printf("¥n");printf("¥n");
      printf("#T-MAN3 S-001 E-%03d¥n",iii);/*画面表示*/
      printf("¥n");
      for (i=1;i<=iii;i++)
       {    /* 配列から読む */
         fpx=dtzpnt[i].x;
         fpy=dtzpnt[i].y;
/* ファイルへ書き込む */
      fprintf(fpo,"T1 PT-%03d X-+%07.2f Y-+%07.2f Z-+0000.00
 R-+0000.00¥n",i,fpx,fpy);
      printf("T1 PT-%03d X-+%07.2f Y-+%07.2f Z-+0000.00
 R-+0000.00¥n",i,fpx,fpy);/*画面表示*/
        }
        fclose(fpo);
        }
    printf("¥33[36m");
    printf("¥n%s WAS CREATED !!¥n",tpname);
    printf("¥33[m");printf("¥n");printf("¥n");printf("¥n");
}
/*************************************************************
*                          *.sq3ファイルへの書き込み          *
*************************************************************/
void sqwrite(char* sqname)
{
    FILE *fpo;
    fpo = fopen(sqname,"w");
    if (fpo == NULL)
    {
        printf("¥33[41m");
        printf("¥nCANNOT OPEN %s!!¥n",sqname);
        printf("¥33[m");
        exit(1);
    }
    else
    {
fprintf(fpo,"#T-MAN3 S-0001 E-0012");
    printf("#T-MAN3 S-0001 E-0012¥n");
fprintf(fpo,"                                    ");
fprintf(fpo,"                                    ¥n");
    printf("¥n");
fprintf(fpo,"0001 SPD V-01¥n");
    printf("0001 SPD V-01¥n");
fprintf(fpo,"0002 CNT CN01-0001¥n");
    printf("0002 CNT CN01-0001¥n");
fprintf(fpo,"0003 OUT PN1=.......0¥n");
    printf("0003 OUT PN1=.......0¥n");
fprintf(fpo,"0004 TAG TAG-001¥n");
    printf("0004 TAG TAG-001¥n");
fprintf(fpo,"0005 IN PN1=.......1¥n");
    printf("0005 IN PN1=.......1¥n");
fprintf(fpo,"0006 OUT PN1=.......1¥n");
    printf("0006 OUT PN1=.......1¥n");
fprintf(fpo,"0007 MOVP a PT-000 CN-01 S V-00 POST¥n");
    printf("0007 MOVP a PT-000 CN-01 S V-00 POST¥n");
fprintf(fpo,"0008 OUT PN1=.......0¥n");
    printf("0008 OUT PN1=.......0¥n");
fprintf(fpo,"0009 CNT+ CN01-0001¥n");
    printf("0009 CNT+ CN01-0001¥n");
fprintf(fpo,"0010 JMPC TAG-001 CN01<=%04d¥n",iii);
    printf("0010 JMPC TAG-001 CN01<=%04d¥n",iii);
fprintf(fpo,"0011 MOV a X-+0000.00 Y-+0000.00 Z-+0000.00
 R-+0000.00 S V-00 POST¥n");
    printf("0011 MOV a X-+0000.00 Y-+0000.00 Z-+0000.00
 R-+0000.00 S V-00 POST¥n");
fprintf(fpo,"0012 END¥n");
    printf("0012 END¥n");
    }
    fclose(fpo);
    printf("¥33[36m");
    printf("¥n%s WAS CREATED !!¥n",sqname);
    printf("¥33[m");printf("¥n");printf("¥n");
}
/*************************************************************
*      ・配列から読み出し  ・回転角度計算  ・パルス数計算     *
*      ・.pmdファイルへ格納      ・-π～π → 0～2π            *
*************************************************************/
void pmdwrite(char *pmdname)
{
FILE *fpo;
int abs(int);
float absang;
float fpx;
int fpy;
```

図12－2　デジタイザからのデータを各種フォーマットに加工するためのプログラム

```
        int i;
        /* データ個数の読み取り */
            pntno=dtzpnt[0].y;
            pmdpnt[0].n=pntno;
        /* 既定データ数終了？ */
            while ( pntcnt <= pntno )
            {
        /* 座標読み取りデータ読み出し */
                xx2=dtzpnt[pntcnt].x;
                yy2=dtzpnt[pntcnt].y;
        /* 回転角度計算 */
                xx=(xx2-xx1);
                yy=(yy2-yy1);
                ang2=atan2(yy,xx)*180/3.14159265;
                ang=ang2-ang1;
                absang=fabs(ang);
                if (absang > 180 )/* (-π ～ π)  →  (0 ～ 2π) */
                    ang=ang+360;
                    pulsesuu=(ang+0.05)*10;
                    pulsesuu=abs(pulsesuu);
                    pmdpnt[pntcnt].m=ang;
                    pmdpnt[pntcnt].n=pulsesuu;
                    xx1=xx2;
                    yy1=yy2;
                    ang1=ang2;
        /* 座標指示カウンターインクリメント */
                    pntcnt++;
            }
        fpo = fopen(pmdname,"w");
        if (fpo == NULL)
        {
             printf("33[41m");
             printf("¥nCANNOT OPEN %s!!¥n",pmdname);
             printf("¥33[m");
             exit(1);
        }
        else
        {
            printf("¥n");printf("¥n");
            for (i=0;i<=pntno;i++)
               {  /* 配列から読み、ファイルへ書き込む */
                    fpx = pmdpnt[i].m;
                    fpy = pmdpnt[i].n;
                    fprintf(fpo,"%f,%d¥n",fpx,fpy);
                    printf("%10f,%4d¥n",fpx,fpy);
               }
            fclose(fpo);
        }
        printf("¥33[36m");
        printf("¥n%s WAS CREATED !!¥n",pmdname);
        printf("¥33[m");
    }
```

図12－3　デジタイザからのデータを各種フォーマットに加工するためのプログラム

```
/*************************************************************
*DRGRPH.C              *. d t z をもとに描画する              *
*************************************************************/
#include        <stdio.h>
#include        <graph.h>
#include        <string.h>
#include        <math.h>
#include        <stdlib.h>
int ox=0,oy=0,dtzx,dtzy,dtzn=1;
int ii,iii,fa;
int pntno;/*座標データ個数*/
float xa,ya,xb,yb,xc,yc;/*配列からのX,Yデータ*/
char name[80];
char filename[80];
const char *d_drive = "d:";
char dtzname[80];
void dtzread(char *);
struct dtzpoint
    {    float x;
         float y;
    }    dtzpnt[512];

int main()
{
    _setvideomode(_98RES16COLOR);/*ビデオモードの設定*/
    _setcolor(11);/*画面の色彩設定*/
    _rectangle(_GFILLINTERIOR,0,0,639,399);/*画面の作成*/
    printf("¥33[2J");/*テキスト画面のクリア*/
    strcpy(filename,d_drive);
    printf("¥33[33m");/*テキストカラーの設定*/
    printf("INPUT FILE NAME(*.dtz)¥n");/*ファイル名の入力*/
    printf("¥33[m");/*テキストカラーを元に戻す*/
    scanf("%s",name);
    strcat(filename,name);
    strcpy(dtzname,filename);
    strcat(dtzname,".dtz");
    dtzread(dtzname);
    printf("¥33[32m");
    printf("¥nTOTAL %d POINTS¥n",ii);/*ポイントの個数*/
    printf("¥33[m");
    printf("¥n");
    for (iii=1;iii<=ii;iii++)
        {    /* 座標読み取りデータ読み出し */
        xa=dtzpnt[iii].x;
        ya=dtzpnt[iii].y;
        printf("P%2d (%7.2f ,%7.2f )¥n",iii,xa,ya);
        }
    printf("¥33[33m");
    printf("¥nINPUT CANVAS SIZE¥n");
    printf("¥nSIZE=");printf("¥33[m");
    scanf("%d",&dtzx);
    dtzy=-dtzx;
    _setcolor(15);/*描画エリアの色彩設定*/
    _rectangle(_GFILLINTERIOR,250,5,634,389);/*描画ｴﾘｱの作成*/
    _setviewport(250,5,634,389);/*ビューポートの設定*/
    _setvieworg(250,389);/*ビューポートの原点設定*/
    _setwindow(1,ox,oy,ox+dtzx/dtzn,dtzy/dtzn);/*ｳｲﾝﾄﾞｳの設定*/
        for (iii=1;iii<=ii-1;iii++)
        {
          xb=dtzpnt[iii].x;
          yb=dtzpnt[iii].y;
        _setcolor(0);
        _moveto_w(xb,yb);
        _ellipse_w(_GFILLINTERIOR,xb-3,yb-3,xb+3,yb+3);
          xc=dtzpnt[iii+1].x;
          yc=dtzpnt[iii+1].y;
        _lineto_w(xc,yc);
        _ellipse_w(_GFILLINTERIOR,xc-3,yc-3,xc+3,yc+3);
        }
  getch();
  _clearscreen(_GCLEARGRAPH);
  _setvideomode(_DEFAULTMODE);
  return 0;
}
/*************************************************************
*              *.dtzファイルから読み出し、配列に格納          *
*************************************************************/
void dtzread(char *dtzname)
{    /*ファイルからの入力*/
    FILE *fp;
    float fpx,fpy;
    int i,fpf;
    fp = fopen(dtzname,"r");
    if (fp == NULL)
    {
        printf("¥33[41m");
        printf("¥nCANNOT OPEN %s!!¥n",dtzname);
        printf("¥33[m");
        getch();
        _clearscreen(_GCLEARGRAPH);
        _setvideomode(_DEFAULTMODE);
        exit(1);
    }
    else
    {
        i = 0;
        while (fscanf(fp," %f , %f ",&fpx,&fpy) !=EOF)
        {    /*配列へ入れる*/
            dtzpnt[i].x=fpx;
            dtzpnt[i].y=fpy;
            i++; ii=i-1;
        }    fclose(fp);
    }
}
```

図13　ディスプレイ画面上へ出力するためのプログラム