**專題實驗報告**

**嵌入式數位元影像預處理硬體加速器設計與實作**

**Design and Implementation of Embanking Hardware Accelerator for Digital Image Preprocessing**

指導教授：陳慶翰

學生：閆明 104502910 [drynessgarden495@gamil.com](mailto:drynessgarden495@gamil.com)

**國立中央大學**

**資訊工程系**

中華民國一〇五年六月

目錄

[摘要 4](#_Toc27472)

[ABSTRACT 5](#_Toc22154)

[第一章 、緒論 6](#_Toc29453)

[1.1 問題描述 6](#_Toc10466)

[1.2 研究目的 7](#_Toc13176)

[1.3 論文架構 7](#_Toc25826)

[第二章 、原理與方法 8](#_Toc7376)

[2.1 預處理 9](#_Toc4467)

[2.1.1 去除雜訊(Denoising) 9](#_Toc7026)

[2.1.2 二值化(Threshold) 12](#_Toc20015)

[2.1.3 邊緣檢測(Edge Detection) 13](#_Toc27260)

[2.1.4 傾斜校正(Tile Correction) 17](#_Toc17406)

[2.1.5 文字分離(Character Segmentation) 21](#_Toc2156)

[2.2 特徵提取與建立範本 21](#_Toc29622)

[2.3 匹配與辨識 24](#_Toc17868)

[第三章 、系統設計 25](#_Toc6810)

[3.1系統架構設計(System Architecture Design) 26](#_Toc26159)

[3.1.1 發票裁剪系統架構設計 28](#_Toc25577)

[3.2 離散事件建模 30](#_Toc19966)

[3.2.1 發票裁剪系統離散事件建模 33](#_Toc6115)

[第四章 、軟硬體合成 38](#_Toc19446)

[4.1 票據剪裁系統軟體合成 39](#_Toc15337)

[第五章 、系統整合驗證 43](#_Toc17215)

[5.1 實驗平台 43](#_Toc29856)

[5.2 驗證方法 45](#_Toc25472)

[5.3 系統驗證 45](#_Toc16969)

[5.4 實驗結果比較與分析 46](#_Toc18020)

[第六章 、結論 47](#_Toc12451)

[Reference 49](#_Toc7957)

**摘要**

光學字元辨識系統(OCR)在機器視覺中扮演了重要的角色，而影像預處理是數位元影像處理不可或缺的一部分。影像預處理是在影像處理分析前，將影像中的雜訊過濾刪除，或者增強影像部分特徵的工作。傳統的數位元影像預處理演算法包括二值化(Thresholding)、消除雜訊(Noise Reduction)、影像裁切(Image Cut)、傾斜校正(Tile Correction)、邊緣檢測(Edge Detection)等，需要佔用大量運算資源，花費大量時間做矩陣運算，導致在資源受限的嵌入式軟體系統中運用影像處理演算法效率低下。

本專題主要針對發票影像的裁剪與文字辨識實作，過程主要分三個階段：影像輸入與硬體預處理階段、編碼通訊階段、文字辨識階段。在預處理階段，先通過影像傳感器獲取影像資訊，將資訊傳送進現場可程式邏輯門陣列(FPGA)，對影像做二值化、消除雜訊、邊緣檢測、傾斜校正與裁剪預處理。在編碼通訊階段，被預處理後的影像進行壓縮編碼，以輕量級資料交換語言格式傳送至PC端，由PC端進行文字辨識。在文字辨識階段，基於訓練的範本資料進行模式匹配，並將匹配結果返回至嵌入式系統。

ABSTRACT

Preprocessing played a necessary role on Digital Image Process.Distorted text lines,skew,noise and other defects is commonly used in Preprocessing on the front of formal processing.Traditional preprocessing algorithm has Threshoding,Noise Reduction,Image Cut, Tile Correction,Edge Detection and more.Due to typical preprocessing algorithm needs high efficiency resource,need mass reversing matrix calculation time to reduce the proseccing.It is low efficiency to implement a preprocessing algorithm in resource-constrained embeding system.

In this project, we design and implement a high efficiency architecture for cut bill and character recognition accelerators.The architecture of the accelerator contains image input and preprocessing module, formating and transmission module and text recognition module.Processing module get the image information from smart sensor camera and use them as the template of preprocessing, executed threshoding,noise removing, edge detecion and tile correction on FPGA.On transmisson module,construct human-readable text to transmit preprossed image data to PC terminal.Character recognition module based on pattern matching and return back the result to emeding system.

1. **、緒論**

**1.1 問題描述**

近年來，由於影像處理技術的進步與視訊系統(Video Systems)的普及，針對發票以及常見的表單、票據等紙質文檔數據掃描識別的系統，利用OCR識別技術，對發票掃描識別，快速完成表單、票據的資訊採集票據掃描識別系統漸漸完善。此項服務可與政府、企事業單位、工商、保險等多個行業的業務流程系統無縫結合，輔助辦公人員進行增值稅專用發票掃描識別等單據的資訊錄入，提高資料電子化、數據格式化的效率。

由於實際系統中實時(real-time)性能要求較高，在軟體上實施傳統的影像處理架構往往會由於處理時間較長，難以滿足系統的要求。現場可程式邏輯門陣列(FPGA)在過去10多年的技術發展,給數位訊號處理(Digital Signal Process)演算法帶來了一條全新的設計方案。FPGA保持了特殊應用積體電路(Application Specific Integrated Circuit) 的高速性,又避免了研發的高成本以及再成品後不能改變電路的特點。其本身的強大並行處理能力(concurrent processing)適合細細微性(fine grain)以及高並行度(high concurrent)為結構特點的數位訊號處理任務[1]。[[1]](#footnote-0)因此通常採用FPGA作為影像預處理的加速器。

**1.2 研究目的**

本論文將探討對影像的預處理、OCR辨識的原理，進而提出一個高性能的影像處理軟硬體架構，並設計一個滿足實時性嵌入式系統需求的嵌入式硬體加速器與OCR辨識系統。在預處理方面，使用自適應大津演算法(OSTU cvAdaptive Threshold)作為影像二值化處理演算法，採用中值濾波器(Median Filtering)對輸入影像做去除雜訊處理(Denoising)，採用canny運算元對去雜訊影像做邊緣提取，採用霍夫變換(Hough Transform)對邊緣提取影像做糾偏處理(Skewness)。該架構嚴格遵守MIAT方法論，對上述方法進行階層式、模組化分析，並做離散事件建模，最後做軟硬體整合。

**1.3 論文架構**

本論文第二章將探討目前的主流影像預處理演算法和OCR技術原理。第三章介紹基於MIAT方法論，對整個解決方案用IDEF0做階層式、模組化切割，再用Grafcet做離散事件建模。第四章依照第三章的方法論做相應的軟硬體整合。第五章做系統整合驗證。第六章提出結論與心得。

1. **、原理與方法**

OCR （Optical Character Recognition）是指電子設備（例如掃描器或數碼相機）檢查紙上列印的字元，通過檢測暗、亮的模式確定其形狀，然後用字元識別方法將形狀翻譯成電腦文字的過程；即，針對印刷體字元，採用光學的方式將紙質文檔中的文字轉換成為黑白點陣的圖像檔，並通過識別軟體將圖像中的文字轉換成文本格式，供文字處理軟體進一步編輯加工的技術[2]。[[2]](#footnote-1)

一個基於範本匹配(Template Matching )的OCR辨識系統的演算法架構如下圖2.1：

測試資料

應用資料

預處理

特徵提取

建立模板

預處理

分類辨識

特徵提取

Matching

Matching

圖2.1 基於範本匹配的OCR辨識系統的演算法架構

**2.1 預處理**

對影像進行預處理以便後續進行特徵提取、學習。主要目的是為了減少影像中的雜訊。在這個步驟通常有：灰度化(Grayscale)、二值化(Threshold)、去除雜訊(Denoising)、傾斜校正(Tile Correction)、邊緣檢測(Edge Detection)、旋轉(Rotation)、文字分離(Character Segmentation)等步驟。經過二值化後，圖像只剩下兩種顏色，即黑和白，其中一個是圖像背景，另一個顏色就是要識別的文字了。降噪在這個階段非常重要，降噪演算法的好壞對特徵提取的影響很大。文字分離則是將圖像中的文字分割成單個文字——識別的時候是一個字一個字識別的。如果文字行有傾斜的話往往還要進行傾斜校正。

**2.1.1 去除雜訊(Denoising)**

去除雜訊技術屬於影像增強技術(Image Enhancement)與影像復原技術(Image Restoration)的相交叉的領域。是一個同時帶有主觀過程與客觀過程的處理。

雜訊(Noise)，指當訊號在傳輸過程中，會受到一些外在能量所產生訊號（如雜散[電磁場](https://zh.wikipedia.org/wiki/%E7%94%B5%E7%A3%81%E5%9C%BA" \o "電磁場)）幹擾，這些能量即雜訊。雜訊通常會造成訊號的失真。其來源除了來自系統外部，亦有可能由接收系統本身產生。雜訊的強度通常都是與訊號[頻寬](https://zh.wikipedia.org/wiki/%E9%A0%BB%E5%AF%AC" \o "頻寬)成正比，所以當訊號頻寬越寬，雜訊的幹擾也會越大。所以在評估雜訊強度或是系統抵抗雜訊能力的數據，是以訊號強度對雜訊強度的比例為依據，此即[訊雜比](https://zh.wikipedia.org/wiki/%E8%A8%8A%E9%9B%9C%E6%AF%94" \o "訊雜比)[3]。[[3]](#footnote-2)

基於雜訊的分量灰度值的統計特性(the statistical behavior of the gray-level values)，它們可以被認為是由概率密度函數(Probablility Density Function)而表示的隨機變數。以此為標準，可以有如下分類：

**高斯噪聲(Gaussian Noise)**

高斯噪聲是一種具有[常態分佈](https://zh.wikipedia.org/wiki/%E6%AD%A3%E6%80%81%E5%88%86%E5%B8%83" \o "常態分布)（也稱作[高斯分佈](https://zh.wikipedia.org/wiki/%E9%AB%98%E6%96%AF%E5%88%86%E5%B8%83" \o "高斯分布)）[機率密度函數](https://zh.wikipedia.org/wiki/%E6%A6%82%E7%8E%87%E5%AF%86%E5%BA%A6%E5%87%BD%E6%95%B0" \o "機率密度函數)的[噪聲](https://zh.wikipedia.org/wiki/%E9%9B%9C%E8%A8%8A_(%E9%80%9A%E8%A8%8A%E5%AD%B8)" \o "雜訊 (通訊學))。換句話說，高斯噪聲的值遵循高斯分佈或者它在各個頻率分量上的能量具有高斯分佈。它被極其普遍地應用為用以產生[加成性高斯白噪聲](https://zh.wikipedia.org/wiki/AWGN" \o "AWGN)（AWGN）的疊代[白噪聲](https://zh.wikipedia.org/wiki/%E7%99%BD%E5%99%AA%E5%A3%B0" \o "白噪聲)[4]。[[4]](#footnote-3)

高斯隨機變數z的PDF如下式:

（公式 2.1）

其中，z表示灰度值，μ表示z的期望值，σ表示z的標準差(下同)。當z服從該式時，其值有約70%落在[(μ-σ),(μ+σ)]範圍內。其值有95%落在[(μ-2σ),(μ+2σ)]範圍內

**瑞利噪聲(Rayleigh Noise)**

瑞利噪聲的PDF如下式：

** （公式2.2）**

瑞利噪聲的噪聲分佈基本向右偏移。

**伽馬噪聲(Erlang(Gamma) Noise)**

伽馬噪聲的PDF如下式：

**（公式2.3）**

其中a、b均為正整數。

**椒鹽噪聲(Salt-and-Pepper Noise)**

椒鹽噪聲的PDF如下式：

**（公式2.4）**

其中，如果b>a，灰度值b在影像中為亮點，否則為暗點。

本論文採用中值濾波器作為去除雜訊演算法：

**中值濾波器(MidIan Filter)**

中值濾波器是最著名的順序統計濾波器(Order-statistics Filter)，使用該圖元的中值來替代該圖元的值[5]。[[5]](#footnote-4)中值濾波器有良好的去除雜訊的能力，有效去除上述噪聲，尤其是椒鹽噪聲。

一個數值集合的中值ζ滿足：在數值集合中，有一半小於或等於ζ，還有一半大於或等於ζ。對一個範本(imtemplate)的圖元(pixel)做排序(sorting)，確定出中值，並將該中值賦予該圖元點。即：

**（公式2.5）**

式中是疊加了雜訊的輸入影像，是中值濾波後的輸出影像。 範本寬度的一半必須大於雜訊點的寬度。

範本通常可選擇為3X3,5X5,7X7等。例如，對於一個3X3的鄰域(neighorhood),中值為5，對於一個5X5的鄰域(neighorhood),中值為13。

**2.1.2 二值化(Threshold)**

若目標和背景具有不同的灰度集合，且兩個灰度集合可用一個灰度級閾值T進行分割，即二值化。

**（公式2.6）**

二值化是[分割](https://zh.wikipedia.org/wiki/%E5%9B%BE%E5%83%8F%E5%88%86%E5%89%B2" \o "圖像分割)的一種方法。在二值化影像的時候把大於某個臨界灰度值的圖元灰度設為灰度極大值，把小於這個值的圖元灰度設為灰度極小值，從而實現二值化。

根據閾值選取的不同，二值化的演算法分為固定閾值和自適應閾值。 比較常用的二值化方法則有：雙峰法、P參數法、疊代法和[OTSU法](https://zh.wikipedia.org/wiki/%E5%A4%A7%E6%B4%A5%E7%AE%97%E6%B3%95" \o "大津算法)等[6]。[[6]](#footnote-5)

本論文採用OTSU演算法。

**最大類間方差法(OTSU)**

假設T為選取閾值，則利用灰度閾值T對物體的面積進行運算的定義是：

**（公式2.7）**

當取最佳閾值時，背景應該與前景差別最大，關鍵在於如何選擇衡量差別的標準。

而在OTSU演算法中這個衡量差別的標準就是最大類間方差，即 ：以最佳閾值將影像的灰度直方圖(Histogram)分割成兩部分，是兩部分之間的方差取最大值，即分離性最大。

因此該演算法也是在灰度直方圖的基礎上用最小二乘法(Generalized Least Squares)原理推導出來的，具有統計意義上的最佳分割閾值，錯分面積最小。

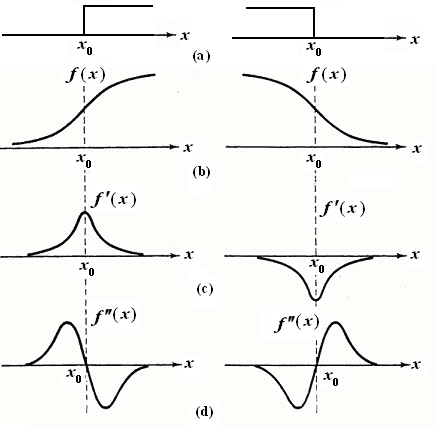
**2.1.3 邊緣檢測(Edge Detection)**

在數位元影像處理中，邊緣檢測和二值化都屬於影像切割(Image Segmentation)的範疇，不同之處在於邊緣檢測是一種改進全域閾值處理的策略。

**邊緣(Edge)**

圖像的邊緣是指圖像局部性質（如灰度、顏色、紋理等）發生明顯變化的邊界或輪廓。

灰度圖像的邊緣可以用微分運算元(Differential Operator)檢測出來。圖像的邊緣位置對應於一階導數的模極大值所在位置和二階導數的零交叉位置，如下圖。

[[7]](#footnote-6)

圖出自[7]

圖2.2圖像的邊緣位置對應於一階導數的模極大值所在位置和二階導數的零交叉位置

邊緣檢測是基於灰度不連續性(Discontinuous Gray Image)進行的切割方法。差分、梯度、拉普拉斯運算元及各種高通濾波(High Pass Filter)處理方法對影像邊緣進行增強，只要再進行一次門限化(Threshold)的處理，便可以將邊緣增強的方法用於邊緣檢測。

**梯度運算元(Nabla Operator)**

圖像的梯度定義為

 （公式2.8）

它是一個向量，其模和方向角分別是：

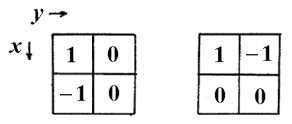
（公式2.9）

 （公式2.10）

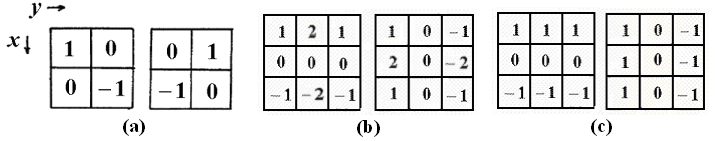
數位元影像是離散信號，偏導數可以用差分來逼近，轉化為 為簡化計算，上式還可以進一步簡化為：



由上式可以看出，離散梯度運算元可以用所示的兩個範本來實作。



基於離散梯度的常用邊緣運算元有Roberts運算元、Sobel運算元、Prewitt運算元等，它們的形式分別如下圖(a)、(b)、(c)所示。這些運算元是有方向的，一般需要兩個(或多個)範本檢測不同方向的邊緣。

[[8]](#footnote-7)

1. Roberts ; (b)Sobel; (c) Preeitt

圖出自[8]

邊緣檢測過程如下圖所示。圖中T為設定的閾值，若點的梯度值大於，則該點是邊緣點，否則該點不是邊緣點。輸出訊號是一幅二值邊緣圖，即

（公式2.11）

**Laplace運算元（零交叉法）**

拉普拉斯運算元是一種基於二階導數(Second Dirvative)的邊緣檢測運算元，其數學運算式為

（公式2.12）

對於數字影像，需要導出上式的差分形式:





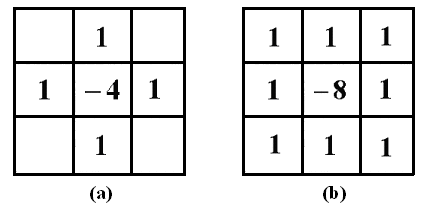
同樣



可得



可以看出，拉普拉斯運算元可以用下圖所示範本與影像的卷積運算來實現。圖(b)是一個更為常用的拉普拉斯運算元範本。

[[9]](#footnote-8)

圖來自[9]

拉普拉斯運算元是無方向資訊的標量運算元，它只需要一個範本就可以進行邊緣檢測。它是一個二階導數運算元，其零交叉位置就是邊緣位置。不過二階導數對雜訊很敏感，用它進行邊緣檢測會出現假輪廓。

**2.1.4 傾斜校正(Tile Correction)**

傾斜校正在本論文中並非指代畸變校正(Correction of Distortion)，下同。而事實上，本文所指代的傾斜校正也屬於影像裁剪(Image Enhancement)的領域的技術。不同之處在於傾斜校正是邊緣連接(Edge Linking)部分。如果邊緣很明顯，而且雜訊級低，那麼可以將邊緣圖像二值化並將其細化為單圖元寬的閉合連通邊界圖。然而，雜訊和不均勻的照明而產生的邊緣間斷的影響，使得經過邊緣檢測後得到的邊緣圖元點很少能完整地描繪實際的一條邊緣。

由於本論文面向的發票圖片主體具有矩陣特性，可以用Hough變換對發票的矩形區域的上下和左右平行邊界進行查找。

**Hougu變換(Hough Transform)**

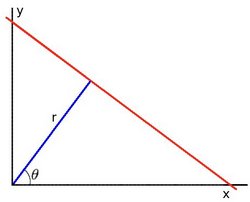
Hough變換是通過運用兩個坐標系之間的轉換來檢測出平面內的直線與有規律的曲線，這種變換具有在變換域的邊緣區有凝聚成峰點的特性。

Hough變換將笛卡爾坐標系((Cartesian coordinates)轉化為極坐標(polar coordinates)系之後,原始坐標系的曲線與直線特徵變換為點特徵，因此原始坐標系的給定形狀的直線與曲線將凝聚為極坐標系的峰點。通過尋找峰點，來檢測整體特徵(原始坐標空間中的給定曲線或者直線的點集特徵)轉換為檢測局部特徵(Hough 空間中點的特徵)。

然而由於Hough變換本身的局限性，它只能對曲線形狀用參數曲線方程(如直線、二次曲線)來描述，而無法對曲線的位置做出描述

**經典 Hough 變換**  
 一條直線在圖像二維空間可由兩個變數表示。例如：

* 1. 在 笛卡爾坐標系： 可由參數 IMG_256 斜率和截距表示
  2. 在 極坐標系：可由參數IMG_257 極徑和極角表示



對於霍夫變換，我們將用極坐標系來表示直線。因此，直線的運算式可為：

IMG_259（公式2.13）

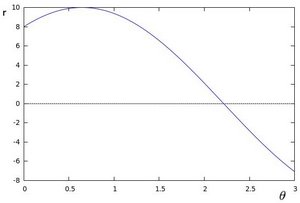
化簡得IMG_260

一般來說對於點，我們可以將通過這個點的一族直線統一定義為：

IMG_262

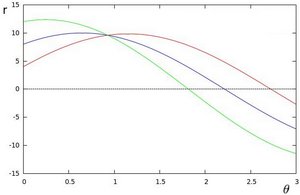
這就意味著每一對 IMG_263 代表一條通過點的直線。

如果對於一個給定點，我們在極座標對極徑極角平面繪出所有通過它的直線，將得到一條正弦曲線。例如，對於給定點 IMG_266and 我們可以繪出下圖 (在平面 IMG_268 - IMG_269)：



只繪出滿足下列條件的點 IMG_271 and IMG_272。

我們可以對圖像中所有的點進行上述操作。如果兩個不同點進行上述操作後得到的曲線在平面 IMG_273 - IMG_274 相交，這就意味著它們通過同一條直線。例如，接上面的例子我們繼續對點： IMG_275， IMG_276 和點 IMG_277，IMG_278 繪圖，得到下圖：



這三條曲線在 IMG_280 - IMG_281 平面相交於點 IMG_282，座標表示的是參數對 (IMG_283) 或者是說點 IMG_256，點 IMG_285 和點 IMG_286 組成的平面內的的直線。

一條直線能夠通過在平面 IMG_287 - IMG_288 尋找交於一點的曲線數量來檢測。越多曲線交於一點也就意味著這個交點表示的直線由更多的點組成。一般來說我們可以通過設置直線上點的閾值來定義多少條曲線交於一點我們才認為檢測 到了一條直線。

追蹤圖像中每個點對應曲線間的交點。 如果交於一點的曲線的數量超過了 閾值，那麼可以認為這個交點所代表的參數對 IMG_289 在原圖像中為一條直線。

**2.1.5 文字分離(Character Segmentation)**

文字分離屬於影像切割(Image Segmentation)的範疇，是全域閾值分割和自適應閾值分割技術。圖像的邊緣位置對應於一階導數的模極大值所在位置和二階導數的零交叉位置。將已二值化圖像向水準方向投影，這樣有字的地方的投影值就高，沒字的地方投影得到的值就低。

（公式2.14）

 然後，用一根掃描線從下向上掃描。這個掃描線會與圖中曲線存在交點，設定閾值T，這些交點會根據T的值將分割成一個又一個區域。。然後根據這些區域向水準線的投影的座標就可以將圖片字元分割出來。

以上是平行投影。然後還需要做垂直投影與切割。

**2.2 特徵提取與建立範本**

特徵是用來識別文字的關鍵資訊，每個不同的文字都能通過特徵來和其他文字進行區分。對於數字和英文字母來說，這個特徵提取是比較容易的，因為數字只有10個，英文字母只有52個，都是小字元集。對於漢字來說，特徵提取比較困難，因為首先漢字是大字元集，國標中光是最常用的第一級漢字就有3755個；第二個漢字結構複雜，形近字多。在確定了使用何種特徵後，視情況而定，還有可能要進行特徵降維，這種情況就是如果特徵的維數太高（特徵一般用一個向量表示，維數即該向量的分量數），分類器的效率會受到很大的影響，為了提高識別速率，往往就要進行降維，又得使得減少維數後的特徵向量還保留了足夠的資訊量（以區分不同的文字）[10]。[[10]](#footnote-9)

對已分割好的字元影像的要求有以下三點：

1. 保留字元特徵(如:‘A’與’B’具有明顯不同的特徵)
2. 放縮不變性(Scaling Immutability)
3. 易於維修與更新

傳統的特徵提取方法有基於幾何距不變性(Moments Immutability)特徵提取、二值圖像輪廓提取(Contour Extraction)、鏈式編碼(Chain Code)等。

**幾何距(Moment)特徵提取**

基於空間Moments演算法在圖像處理與分析中尋找連通區域計算連通區域的中心與角度方向Moments的一階可以用來計算區域的中心質點，二階可以用來證明圖像的幾個不變性。如旋轉不變行，放縮不變性等。基於Moments的二階計算結果，根據如下公式：

（公式2.15）

可以得到區域的方向角度[11]。[[11]](#footnote-10)

統計圖元的公式如下:



其中，（公式2.16）

計算偏斜度(Skewness)的公式為：

（公式2.16）

根據幾何距不變性原理，計算每個字元的幾何距特徵X1,X2：





或者基於勒讓德距(Legendre Moments)計算幾何距不變性：

**勒讓德距(Legendre Moments)**{\displaystyle {d \over dx}\left[(1-x^{2}){d \over dx}P(x)\right]+n(n+1)P(x)=0.}IMG_257

計算勒讓德距的公式為：

****

則幾何距特徵



其中，Pm和Pn為勒讓德多項式(Legendre Polynomial)，根據勒讓德距計算距不變性詳情請參考加拿大裔華人Simon Liao的著作《Image Analysis by Moments》。

**2.3 匹配與辨識**

匹配與辨識的工作是用來辨識與訓練特徵。對一個字元圖像提取出特徵，輸入給辨識器，辨識器就對其辨識，分析出該字元該被辨識為哪個字元。

1. **、系統設計**

隨著需求的更新，由於系統設計的複雜度漸漸上升，產品的生命週期則是漸漸雖短。如何設計一個同時具備搞複雜度和高功能性有易於維護的系統成為了一項問題。本專題採用MIAT的嵌入式軟硬體設計方法論，來實現整個專題的系統設計，有效降低系統設計與軟硬體整合的複雜度。

MIAT方法論的底層架構如下圖：



圖3.1 MIAT方法論的底層架構

MIAT方法論首先需要在高階語言上驗證演算法功能的正確性。根據系統需求對已驗證演算法基於自頂向下設計方法(Top-Down Paradigm)採用IDEF0作為階層式、模組化的功能結構設計，系統將被分解成一組階層式的獨立模組，並針對每個功能模組使用Grafcet建立離散事件模型，即圖形化行為建模(Graphical Behavior Modeling)，以表達該離散事件系統的輸入輸出與狀態轉移關係。最後按照MIAT的Grafcet與C/VHDL語言的合成規則，將所有Grafcet合成為C/VHDL語言，並進行MCU/FPGA系統整合測試，即可完成整個軟硬體整合架構的高階合成。

**3.1系統架構設計(System Architecture Design)**

採用IDEF0以層級結構化、可視圖形化的方式，將系統功能之間的複雜關係，以及功能有關的物件(Object)與資訊(information)表達出來。IDEF0是由功能方塊(Function Block)與連線箭頭(Arrows)組成，每個功能代表一個系統的活動(Activity)，用以描述系統的過程(Process)、作業(Operation)以及轉換(Transportation)關係，連線箭頭表示功能之間的關係，或描述執行系統功能所需之資訊與物件。連線箭頭的方向意義如下圖所示：

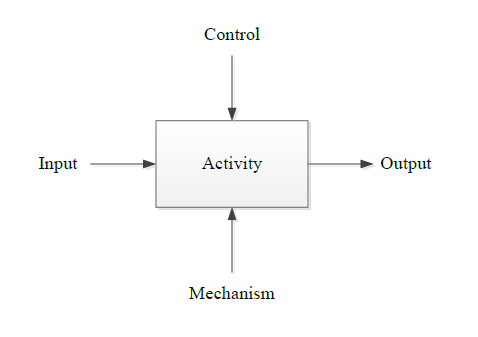


圖3.2連線箭頭的方向意義

通過這種階層化分解，簡化複雜系統的設計。

IDEF0模型是基於自頂向下的設計方式，如下圖：

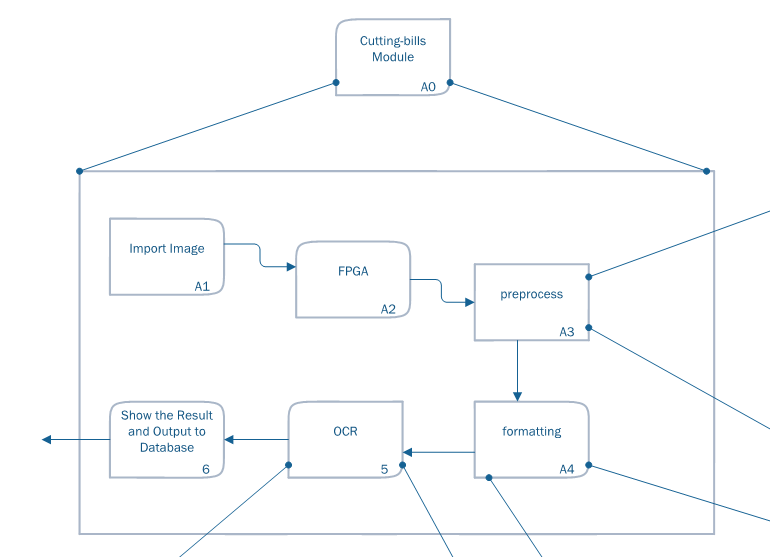


圖3.3 IDEF0模型是基於自頂向下的設計方式

其中，每一組IDEF0模組都是一個獨立的離散事件系統(Discrete Event System，DFS)。按照這種分割方法，可以根據需求變動而做切割延伸下去，以此達到簡化複雜系統設計與維護群體協同(Collective Co-operation)的設計目的。通過這種圖形化的細部分解提高了系統分析的可溝通性，使用簡易的圖形功能分解與學習，使開發者與使用者可以從其階層式擴展架構上建立良好的溝通。

**3.1.1 發票裁剪系統架構設計**

該專題的頂層IDEF0架構如下圖：



Result

Input Image

圖3.4 頂層IDEF0架構

其中，最上層為模組管線化控制器，負責第二層的六個子模組的平行作業。輸入影像為待辨識票據圖片，輸出為辨識結果。A0模組可以再往下切割6個模組，A1為MCU導入影像，A2為基於FMC協定將影像輸入至FPGA，A3為預處理，A4為影像編碼壓縮，A5為文字辨識，A6為輸出結果。

A3模組中，為了增加效能，使用平行化架構設計模組，如圖所示：

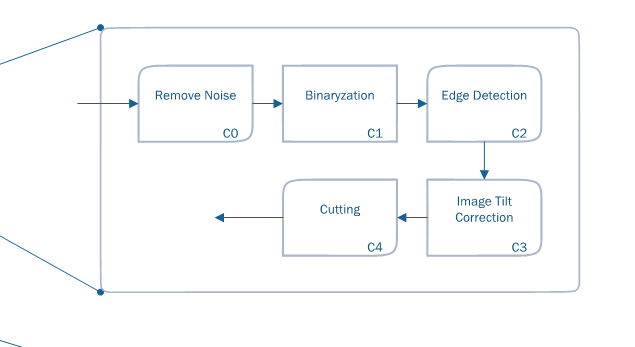


圖3.4 A3模組IDEF0架構

在A3模組中，依次對輸入影像做去除雜訊、二值化、邊緣檢測、傾斜校正、邊框裁剪操作。此過程在FPGA硬體運行。

A模組中，為了增加效能，使用平行化架構設計模組，如圖所示：

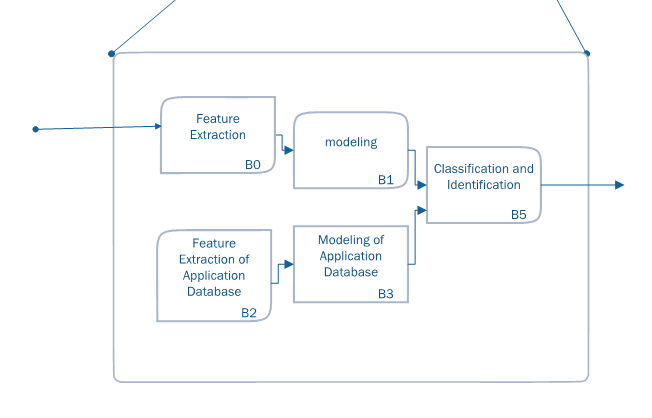


圖3.5 A5模組IDEF0架構

A5模組在中，對原有影像資料做特徵提取，訓練樣本範本，對測試影像資料做匹配與辨識，此過程在PC端完成。

**3.2 離散事件建模**

Grafcet是由Petri nets 發展而來，用來描述連續行為與同時發生情況的一種圖形化模型。GRAFCET標記法，其最初規格是結合學界和工業界的法國團體所定義出來的，並在1987年成為國際標準。在80年代，GRAFCET多使用在事務性機器（OA產品）與製造系統，如自動化元件之控制器，將離散事件控制器設計在可程式控制器（PLC）上；在90年代後期，開始運用在資訊系統設計。GRAFCET可使用在工業製造系統的順序控制規格描述，能夠很簡潔明瞭的表示多輸入與多輸出的同步動作，簡化離散控制邏輯使其易於處理。

Grafcet的基本構件為：

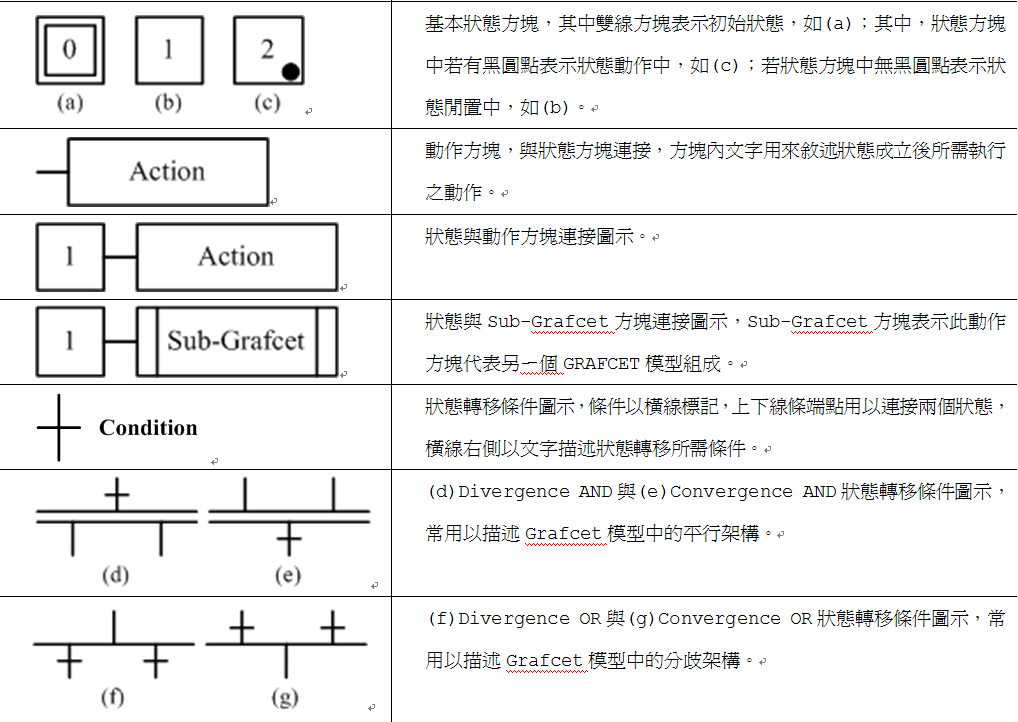
1、步驟(Step)

2、轉移條件(Transition)

3、有向性的連結(Directed Connection)

一個GRAFCET至少包含一個狀態和一個轉移條件以上，連結時由狀態連結到轉移條件或是由轉移條件連結到狀態。。

Grafcet的語法如圖：

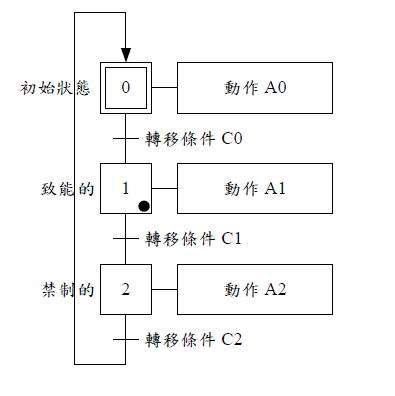


一個狀態轉移的觸發必須符合下列兩個條件：

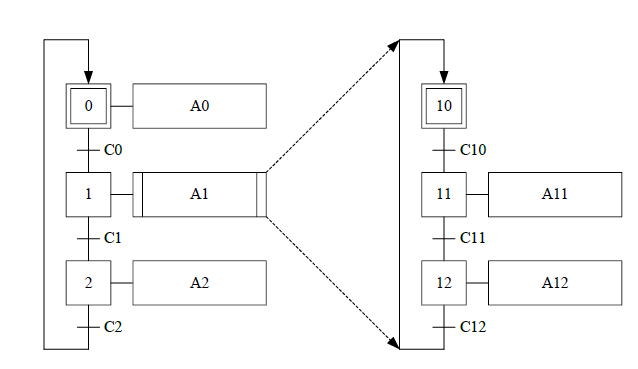
（1）在狀態轉移之前的狀態必須是處於active；

（2）狀態轉移條件為true。

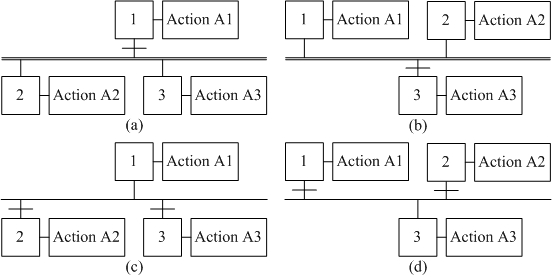
兩個狀態之間只能有一個轉移條件，兩個轉移條件之間只能有一個狀態。當系統過於複雜時，則會以Sub-Grafcet來表示其子階層模組設計。一個Grafcet的基本架構如圖3.6所示：



Sub-Grafcet的描述架構如圖：



並行/分支Grafcet結構如圖：



整個流程結束後，必須回到初始狀態等待下一次的觸發。

Grafcet與一般的演算法描述或有限狀態機(Finite State Machine)的不同點在於，從動作方塊中可以得知需要執行的參數或是邏輯運算，在轉移條件中有提供平行架構，更有效地開發系統。

**3.2.1 發票裁剪系統離散事件建模**

一旦完成系統的IDEF0 階層式模組架構設計，接下來所有模組均使用Grafcet 進行離散事件建模，明確表示所需之輸入、輸出及循序或平行控制流程，Grafcet 模型中的狀態命名是採用方法論中的命名原則，使系統整合更加有效能。如圖所示：



3.6 系統整體Grafcet

Cut-bill System子模組如圖：



圖3.7Cut-bill System子模組Grafcet

Preprocess子模組如圖：



3.8 Preprocess子模組Grafcet

OCR控制器子模組如圖：



圖3.9 OCR控制器子模組Grafcet

Formatiing控制器子模組如圖：



圖3.10 Formatiing控制器子模組

其主要狀態與轉移動作為：

|  |  |  |  |
| --- | --- | --- | --- |
| 狀態 | 動作 | Function Name | 功能說明 |
| 0 | Null | Null | 語者辨識初始狀態 |
| 1 | 資訊擷取 | grafcet1() | 按下按鈕後，儲存由攝像頭接收的資料 |
| 2 | 票據裁剪系統 | Grafcet2() | 開始運行處理系統 |
| 20 | 獲取影像 | TakePohotos() | 讀取攝像頭的資料 |
| 21 | 傳送至FPGA | SendtoFPGA() | 傳送並通知FPGA響應 |
| 22 | 影像預處理 | Preprocessing() | 在FPGA中運行預處理演算法 |
| 23 | 從FPGA讀取已預處理影像 | ReadfromFPGA() | 從FPGA讀取已預處理影像 |
| 24 | 編碼 | Formating() | 對圖像做編碼壓縮 |
| 25 | OCR | Null | 對圖像做文字辨識 |
| 26 | 接收輸出結果 | Null | 輸出OCR的結果 |
| 220 | FPGA初始化 | Null | Do noting |
| 221 | 去除雜訊 | MiddleFilter() | 去除影像源的雜訊 |
| 222 | 二值化 | Threshold() | 對影像做二值化處理 |
| 223 | 邊緣提取 | Tile\_sobel() | 對影像做邊緣提取 |
| 224 | 傾斜校正 | Edge\_Hough() | 對影像做邊緣提取 |
| 225 | 裁剪 | Cutting() | 裁剪票據至票據邊框 |

1. **、軟硬體合成**

Grafcet具明顯的架構，能夠很快的得出對應之軟體語言。MIAT方法論提供了硬體合成規則，能夠將Grafcet模型快速的以c語言合成到MCU上，Grafcet中的每一個狀態和轉移條件可以合成為一段C code，如圖4.1所示。



**void main( )**

**{**

**while(1)**

**{grafcet( );}**

**}**

**void grafcet( )**

**{**

**if((X0==1)&&(R0==1))**

**{X0=0; X1=1;}**

**action();**

**}**

**void action( )**

**{**

**if(x1==1){}**

**if(x2==1){}**

**}**

圖4.1 Grafcet模型合成C code

IDEF0對系統進行階層化和模組化，使得複雜的系統被分解成多個功能模組，在看出主要的功能和流程以後，使用Grafcet進行系統的建模，使系統所有的行為變為Grafcet的流程圖，此圖記錄著每一個狀態及所有邏輯化的動作，再由合成規則進行C語言合成。此法能清晰的看出程式的運行過程，還可以明晰系統中的邏輯判斷及變數的使用，且合成規則有結構性和規則性，使得開發者能夠高效地能的產生程式碼，而且加快了整個系統的開發流程。

**4.1 票據剪裁系統軟體合成**

票據剪裁實作採用圖3.6 的系統架構，分別於A1進行MCU導入影像，於A2進行基於FMC協定將影像輸入至FPGA，A3預處理，A4影像編碼壓縮，A5文字辨識，A6輸出結果，根據方法論合成方法進行合成，合成後代碼如圖4.2 所示。

**軟體合成: main( )**

**int main(void)**

**{**

**SystemClock\_Config();**

**\_\_HAL\_RCC\_GPIOB\_CLK\_ENABLE(); /\* Enable GPIOB clock \*/**

**\_\_HAL\_RCC\_GPIOE\_CLK\_ENABLE(); /\* Enable GPIOE clock \*/**

**HAL\_GPIO\_Init(GPIOB, &GPIO\_InitStruct);**

**GPIO\_InitTypeDef Btn;**

**HAL\_GPIO\_Init(GPIOE, &Btn);**

**HAL\_GPIO\_TogglePin(GPIOB, 13);**

**HAL\_GPIO\_TogglePin(GPIOE, 5);**

**/\* Insert delay 100 ms \*/**

**HAL\_Delay(100);**

**while(1)**

**{**

**grafcet0();**

**}**

**}**

**A1模组Grafcet**

**int grafcet0()**

**{**

**x0=1;**

**if(x0==1&&readPin(13)){x1=1;x0=0;}**

**else if(x1==0&&readPin(5)){x2=1;x1=0;}**

**else if(x2==0&&!readPin(13)){x1=1;x2=0;}**

**else if(x2==0&&readPin(13)){x0=1;x2=0;}**

**action0();**

**}**

**int action0()**

**{**

**if(x0==1){}**

**else if(x1==1)open\_state();**

**else if(x2==1)grafcet1();**

**}**

**对数据剪裁系统进行细分**

**int grafcet1()//cut bill system**

**{**

**x10=1;**

**if(x10==1){x11=1;x10=0;}**

**else if(x11==0){x12=1;x11=0;}**

**else if(x12==0){x13=1;x12=0;}**

**else if(x13==0){x14=1;x13=0;}**

**else if(x14==0){x15=1;x14=0;}**

**else if(x15==0){x16=1;x15=0;}**

**else if(x16==0){x10=1;x16=0;}**

**action3();**

**}**

**int action1()**

**{**

**if(x10==1){}**

**else if(x11==1)Take\_photo();**

**else if(x12==1)grafcet3();//preprocess\_**

**else if(x13==1)read\_FroFPGA();**

**else if(x14==1)formating();**

**else if(x15==1)grafcet4();//orc\_**

**else if(x16==1)output();**

**}**

**预处理模组的Grafcet**

**int grafcet3() //preprocess**

**{**

**x30=1;**

**if(x30==1){x31=1;x30=0;}**

**else if(x31==0){x32=1;x31=0;}**

**else if(x32==0){x30=1;x32=0;}**

**else if(x33==0){x34=1;x33=0;}**

**else if(x34==0){x35=1;x34=0;}**

**else if(x35==0){x36=1;x35=0;}**

**else if(x36==0){x30=1;x36=0;}**

**action3();**

**}**

**int action3()**

**{**

**if(x10==1){}**

**else if(x31==1)check\_fpga();**

**else if(x32==1)Remove\_noise();**

**else if(x33==1)Binaryzation();**

**else if(x34==1)Edge\_detection();**

**else if(x35==1)Image\_tilt\_correction();**

**else if(x36==1)cutting();**

**}**

**圖4.2 軟體合成的c code**

**Orc模组Grafcet**

**int grafcet4() //orc\_**

**{ x40=0;**

**if(x40==1){x40=0;x41=1;x43=1;}**

**else if(x41==1){x41=0;x42=1;}**

**else if(x43==1){x43=0;x44=1;}**

**else if(x42==1&&x44==1&&Data\_ready==1&&Datainfo\_ready==1){x42=0;x44=0;x45=1;}**

**else if(x45==1){x45=0;x40=1;}**

**action4();**

**}**

**int action4()**

**{**

**if(x40==1){}**

**else if(x41==1)Feature\_Extraction();**

**else if(x42==1)modeling();**

**else if(x43==1)Feature\_Extraction\_of\_Application\_Database();**

**else if(x44==1)Modeling\_of\_Application\_Database();**

**else if(x45==1)Classification\_and\_identification();**

**}**

1. **、系統整合驗證**

我們將採用Altera 公司的實驗板來實現本系統，以票據剪裁系統為開發目標。下面將介紹實驗平台，並對系統進行實驗。

**5.1 實驗平台**

本實驗將實作於FPGA 上，由照相機照取一張相片，進入FPGA 後進行運算，最後輸出至PC進行OCR識別，再在螢幕上顯示，由微處理器STM32F446 控制FPGA 運算內容，架構圖如圖5.1 所示。

Camera

OV7725

Max 10 10M08DAF484

PC

STM32F446

圖5.1 實驗環境

本實驗所使用的FPGA(Field Programmable Gate Array)晶片，是採用BeMicro Max 10實驗板來驗證，型號為Max 10 10M08DAF484，此實驗板整合了多個周邊應用介面，其功能表格如表5.1所示，本實驗板可滿足移動視訊、資料存取、影像處理等應用與驗證的需求。

|  |  |
| --- | --- |
| **Item** | **Specification** |
| FPGA device | 8,000 LEs  414 Kbit on-chip memory  256 Kbit user-flash memory  2 PLLs |
| External peripherals | 8MB (4Mbx16) SDRAM  I2C interface  Temperature sensor |

表5.1 MAX 10規格

控制介面STM32F446開發板其核心晶片為32-bit的ARM Cortex-M4 STM32F446ZIT6U，。其系統時脈為180MHz、內含Timer及A/D轉換多組通訊界面，如UART、I2C、SPI等。規格如表5.2所示。

|  |  |
| --- | --- |
| **Item** | **Specification** |
| STM32F429 | 2 MB Flash memory  256 KB RAM  On-board ST-LINK/V2  2.4" QVGA TFT LCD  64 Mbits SDRAM  3-axis digital output gyroscope |

表5.2 STM32F446規格

照相機為smart camera規格如表5.3所示

|  |  |
| --- | --- |
| **Item** | **Specification** |
| Smart  Camera | * MCU: ARM Cortex-M3 STM32F217 * 影像感測器：   + CMOS Sensor * 通訊介面：DCMI、UART * 功能：   + 影像擷取、RGB色彩影像、灰階影像、二值化影像   + 同步平行通訊介面、單一串流通訊介面和高階應用呼叫指令   + 無須撰寫低階感測器函式即可快速應用 |

表5.2 STM32F446規格

**5.2 驗證方法**

由於已將系統各功能做階層式、模組化的設計，因此可以獨立測試各模組，接下來的章節將依序對預處理模組、及整個系統進行驗證。首先預處理部分將使用一張影像，試驗是否能正確進行預處理。接著是驗證OCR可否能夠識別文字再來將整個系統進行試驗，在已知設定好圖片長度和類型的前提下，看是否能在正確的剪裁。

**5.3 系統驗證**

本節將對5.2節所述之步驟進行驗證，首先先針對各模組做效能的評估資料，接著逐步對系統每個模組進行單獨驗證，以此驗證系統整體的可行性。

預處理模組

本節將對系統中的預處理圖形的實驗來驗證最後輸出圖形的正確性。利用320\*240的影像進行偵測實作，實驗內容包括待處理圖像的二值化，去噪，邊緣檢測和旋轉結果，圖5.2為待處理圖像、圖5.4為二值化后圖像、圖5.5為旋轉後圖像，圖5.6為剪裁后圖像。

圖5.2待處理圖像

圖5.4二值化后圖像

圖5.5旋轉後圖像

圖5.6剪裁后圖像。

OCR模組

在PC上進行OCR模組（暫未實現）

**5.4 實驗結果比較與分析**

本論文圖像處理主要分成兩部分，一是進行圖像預處理，以及進行OCR識別并輸出。我們先針對圖像處理在FPGA和PC上進行運行時間上的比較，已同樣的PC平台和嵌入式硬件處理分別以16.53秒及2.43秒作業完成，執行時間大大降低。

1. **、結論**

數位元影像預處理已經應用於生活的許多領域，如指紋識別、車牌識別、自動錄入等，而在數位影像預處理演算法也是多種多樣。本論文先通過影像傳感器獲取影像資訊，將資訊傳送進現場可程式邏輯門陣列(FPGA)，再對圖像進行二值化、消除雜訊、邊緣檢測、傾斜校正與裁剪，對圖像進行了預處理，接著，預處理後的影像進行壓縮編碼，以輕量級資料交換語言格式傳送至PC端，在編碼通訊階段，被由PC端進行文字辨識。在文字辨識階段，基於訓練的範本資料進行模式匹配，並將匹配結果返回。由於傳統的數位元影像預處理演算法包括二值化(Thresholding)、消除雜訊(Noise Reduction)、影像裁切(Image Cut)、傾斜校正(Tile Correction)、邊緣檢測(Edge Detection)等，需要佔用大量運算資源，花費大量時間做矩陣運算，導致在資源受限的嵌入式軟體系統中運用影像處理演算法效率低下，本文中的影像與處理在FPGA內完成，MCU通過FMSC讀取FPGA存儲體裡的內容。

本論文利用了MIAT設計方法論，來降低成本並且讓系統架構邏輯化，先利用IDEF0對系統進行了階層式分析，再使用具有結構化及平行處理能力的Grafcet對每個功能模組進行離散事件行為建模。之後每個模組能各獨立驗證，進一步提升效能，並將預處理模組實現在FPGA上，減少了資源的佔用，大大提升了效率。以期達到具備高速與高效能的目標。

通過學習MIAT設計方法論，使得在系統設計方面有了較為清晰的認識。模組化和階層化較好的分割了系統，使得可以逐一解決和測試。很好的降低了設計系統的成本，也讓人對系統有個十分清晰的認識。

數位影像處理意在降低人工成本並且提高效率和正確性，以嵌入式的方式進行設計系統，有利於之後的高效節能快速的行業發展。

# **Reference**

[1][YE Min](http://xueshu.baidu.com/s?wd=author:(YE Min) Department of Information science and Electronic Engineering, Zhejiang University, Hangzhou 310027, China&tn=SE_baiduxueshu_c1gjeupa&ie=utf-8&sc_f_para=sc_hilight=person" \t "http://xueshu.baidu.com/_blank)，[WH Zhou](http://xueshu.baidu.com/s?wd=author:(ZHOU Wen-hui) Department of Information science and Electronic Engineering, Zhejiang University, Hangzhou 310027, China&tn=SE_baiduxueshu_c1gjeupa&ie=utf-8&sc_f_para=sc_hilight=person" \t "http://xueshu.baidu.com/_blank)，[GU Wei-Kang](http://xueshu.baidu.com/s?wd=author:(GU Wei-kang) Department of Information science and Electronic Engineering, Zhejiang University, Hangzhou 310027, China&tn=SE_baiduxueshu_c1gjeupa&ie=utf-8&sc_f_para=sc_hilight=person" \t "http://xueshu.baidu.com/_blank)，“[FPGA Based Real-Time Image Filtering and Edge Detection](http://xueshu.baidu.com/s?wd=paperuri:(ad8b9156f4349aac58a7a2ce5cec7c0b)&filter=sc_long_sign&tn=SE_xueshusource_2kduw22v&sc_vurl=http://en.cnki.com.cn/article_en/cjfdtotal-cgjs200703034.htm&ie=utf-8&sc_us=11332132544054558626" \t "http://xueshu.baidu.com/_blank)”,[《Chinese Journal of Sensors & Actuators》](http://xueshu.baidu.com/s?wd=journaluri:(cc3422cded2089af) %E3%80%8AChinese Journal of Sensors & Actuators%E3%80%8B&tn=SE_baiduxueshu_c1gjeupa&ie=utf-8&sc_f_para=sc_hilight=publish&sort=sc_cited" \o "《Chinese Journal of Sensors & Actuators》" \t "http://xueshu.baidu.com/_blank), 2007, 20(3):623-627

[2]http://baike.baidu.com/link?url=q9\_OtFZ\_VEi5SgMNSjgfC\_264rv2KYpZyXSqwrQb2Xy7UjbWNdQyabrfbCCNzsyZty3vAmF2CdyeKgAGB1gjPtdgEKEL5UE6HDD7AsknUrPs-03xGlgNyDzXZ9WqRSCM.

[3]https://zh.wikipedia.org/zh-tw/%E9%9B%9C%E8%A8%8A.

[4]https://zh.wikipedia.org/zh-tw/%E9%9B%9C%E8%A8%8A.

[5]Rafael C.Gonzalez, Richard E.Woods[2007] “Digital Image Processing 3rd Edition”PP.185.

[6]https://zh.wikipedia.org/zh-tw/%E4%BA%8C%E5%80%BC%E5%8C%96

[7]Rafael C.Gonzalez, Richard E.Woods[2007] “Digital Image Processing 3rd Edition”PP.475.

[8]Rafael C.Gonzalez, Richard E.Woods[2007] “Digital Image Processing 3rd Edition”PP.469.

[9]Rafael C.Gonzalez, Richard E.Woods[2007] “Digital Image Processing 3rd Edition”PP.471.

[10]http://www.zhihu.com/question/20191727

[11]http://gloomyfish.blog.51cto.com/8837804/1400255

1. FPGA Based Real-Time Image Filtering and Edge Detection. [↑](#footnote-ref-0)
2. http://baike.baidu.com/link?url=q9\_OtFZ\_VEi5SgMNSjgfC\_264rv2KYpZyXSqwrQb2Xy7UjbWNdQyabrfbCCNzsyZty3vAmF2CdyeKgAGB1gjPtdgEKEL5UE6HDD7AsknUrPs-03xGlgNyDzXZ9WqRSCM [↑](#footnote-ref-1)
3. https://zh.wikipedia.org/zh-tw/%E9%9B%9C%E8%A8%8A [↑](#footnote-ref-2)
4. https://zh.wikipedia.org/zh-tw/%E9%AB%98%E6%96%AF%E5%99%AA%E5%A3%B0 [↑](#footnote-ref-3)
5. Digital Image Process P185 [↑](#footnote-ref-4)
6. https://zh.wikipedia.org/zh-tw/%E4%BA%8C%E5%80%BC%E5%8C%96 [↑](#footnote-ref-5)
7. Digital Image Process P475 [↑](#footnote-ref-6)
8. Digital Image Process P.469 [↑](#footnote-ref-7)
9. Digital Image Process P.471 [↑](#footnote-ref-8)
10. http://www.zhihu.com/question/20191727 [↑](#footnote-ref-9)
11. http://gloomyfish.blog.51cto.com/8837804/1400255 [↑](#footnote-ref-10)