論 文
36～8～9

매개변수공간의 동적 분할 방법에 의한 함수패턴의 단계적 분석 추출에 관한 연구

A Study on The Coarse-to-fine Extraction Method of Function Patterns by using The Dynamic Quantization of Parameter Space

*金 民 煥, **黃 熙 隆
(Min-Hwan Kim · Hee-Yeung Hwang)

ABSTRACT

This paper proposes a new method of reducing the processing time and the size of consummimg memories in Hough transform. In this method, only the functional patterns are considered. The candidate points which are accumulated into the parameter space are computed in a many-to-one fashion and the parameter space is quantized dynamically to maintain a fine precision where it is needed. And a coarse-to-fine extraction method is used to reduce the processing time.

The many-to-one fashional computation results in a relatively high-densed accumulation of candidate points around the parameter points corresponding to the image patterns in the image space. So, the dynamic quantization procedure can be simplified and the local maxima can be determined easily. And more effective reduction can be obtained as the dimension of parameter space is increased.

1. 서 론

일반적으로 물체의 특성을 잘 나타내는 것중의 하나로서 경계선은 물체를 인식하는데 대단히 중요한 정보를 제공해 준다. 따라서, 이러한 화상으로부터 의미있는 경계선들을 검출해내는 방법이 많이 연구되어 있다[1]. 그 중의 Hough 변환법은 잡음이나 끊겨진 곡선에 대해서도 별 무리없이 적용시킬 수 있는 것으로서 실제 응용분야에서 널리 사용되고 있는 방법중의 하나이다[2)~5)]. 이것은 매개함수로 표현될 수 있는(parametric) 패턴에 대해 적용시키는 방법과 매개변수로 표현될 수 없는(non-parametric) 패턴에 적용시키는 방법으로 크게 구분될 수가 있는데 본 논문에서는 전자에 관해서만 다루겠다.

Hough 변환법은 화상공간(image space)에 나타나 있는 물체들의 특징점(feature point)들이 갖는 모든 가능한 정보를 매개변수공간(parameter space)에 사상(mapping)시켜 각 정보가 누적되도록 한 후, 나중에 누적된 정보들 중에서 지역적 최대값(local maximum value)을 갖는 것들을 추출함으로써 물체를 찾아내는 방법이다[6),7)].

여기에서 매개변수공간에 누적시킬 정보(이후로 후보점(candidate point)이라고 함.)를 구하는 방법에는 크게 두 가지가 있을 수 있다. 먼저 찾고자 하는 패턴의 매개함수에 하나의 특징점의 좌표값을 대입시켜 얻어진 함수에서 매개변수의 값이 바뀜에 따라 얻어지는 후보점들을 매개변수공간에 누적시키는 방법이 있다. 다른 방법으로는 매개변수 갯수만큼의 특징점들을 조합추출하여 각 조합에 대해 하나의 후보점을 결정하여 누적시키는 방법이 있다[1)].

*正 會 員 : 서울大 工大 電子計算機工學科 博士課程
**正 會 員 : 서울大 工大 電子計算機工學科 敎授 · 工博
接受日字 : 1986年 9월 17日
1次修正 : 1987年 6月 8日
2次修正 : 1987年 7月 13日

전자의 방법은 기존의 Hough 변환법에서 주로 사용되어 왔던 것으로서 화상공간에서의 한 특징점에 대하여 모든 가능한 후보점들을 누적시키는 것이다. 그러나 이러한 후보점들중에서 실제로 존재하는 물체를 나타내기 위한 것은 오직 하나일 뿐이고 나머지는 불필요한 것들이다. 따라서 이러한 불필요한 후보점들이 서로 간섭(중복)을 일으켜 없는 물체를 검출해 내거나 실제로 존재하는 물체를 놓쳐 버리는 결과를 초래하기도 한다[8),9)]. 이 방법에서의 연산량은 각 매개변수가 취할 수 있는 이산값 (discrete value)의 갯수들의 곱에 비례한다[1)].

후자의 방법은 여러 개의 특징점들로부터 얻어질 수 있는 정보를 하나의 후보점으로 표현하여 누적시키는 것이다. 여기에서 서로 관련이 없는 특징점들의 조합에 의해 결정되는 불필요한 후보점들은 매개변수공간에서 산발적으로 흩어져 나타나게 되므로 전자의 방법에서와 같은 불필요한 현상들은 줄어든다. 그러나 연산량은 특징점들의 조합추출 경우의 수에 비례하게 된다. 즉, 특징점 갯수(n)에 매개변수 갯수(d)만큼의 지수승(n^d)에 비례하게 된다[1)].

따라서 특징점의 갯수가 어느 한 매개변수에 대한 가능한 이산값의 갯수보다 많이 나타나는 복잡한 화상에서는 전자의 방법보다 후자의 방법이 더 선호된다[16)].

또한 기존의 Hough 변환법에서는 후보점들을 누적시킬 매개변수공간을 컴퓨터에서 표현하는 방법으로서 배열을 많이 이용하였다. 이러한 배열을 누적배열(accumulator array)이라 하며, 각 매개변수에 대한 이산값의 갯수들의 곱에 해당하는 크기의 메모리 양을 필요로 한다. 이에 따라, 매개변수가 많아질 경우 매우 많은 양의 메모리를 필요로 하게 되어 실제적인 Hough변환의 응용이 매개변수의 갯수가 작은 경우에 한정되는 요인이 되었다[1)]. 이에 대한 해결책으로서 15)에서는 매개변수공간을 고정된 크기로 분할시키는 기존의 방법을 사용하는 대신에 후보점이 많이 누적되는 부분일수록 더욱 세분화된 매개변수공간을 표현하기 위해 k-d 트리를 사용하였다. 또한 각 블럭공간에 누적된 후보점의 갯수를 서로 비슷하게 유지하기 위해 분할(split)뿐만 아니라 합병(merge)시키는 절차를 사용하였다. 그러나 합병절차가 매우 복잡하고 어려운 단점이 있다[14)]. 이에 비해, 14)에서는 소요메모리의 감소효과는 떨어지지만 보다 간단한 절차로 동적분할하는 방법을 사용하였다.

본 논문에서는 14)에서 제안된 매개변수공간의 동적분할 방법을 적용시켜 보았다. 이때, 여러 개의 특징점들을 조합추출하여 하나의 후보점을 결정하는 방법을 사용함으로써 동적분할 과정을 간단하게 하였다. 또한 먼저 화상의 해상도를 낮추어 특징점의 갯수를 줄인 상태에서 개략적인 패턴분류를 행하여 각 후보 패턴을 구한 후, 다음에는 각 후보 패턴의 본래의 화상에서의 특징점들에 대해 다시 변환법을 적용하여 실제의 패턴들만을 확인 추출하였다.

2. 전반적인 절차

이 논문에서 다루고 있는 전반적인 절차는 그림 1과 같다. 먼저 (a)와 같이 입력된 다치 화상(grey image)에 edge연산자를 적용시켜 (b)와 같이 경계선만 나타나도록 한다. 여기에서 만약 입력된 화상의 해상도가 높으면, (b)에 나타나는 특징점들이 너무 많아져 이들 상호간의 조합에 의한 후보점을 계산하는데 너무 많은 연산이 필요하게 된다. 그러나 여기에서 서로 다른 패턴에 속해 있는 특징점들의 조합은 불필요한 후보점을 만들어 내며 연산시간의 증가현상만을 초래한다. 예를 들어, 그림 1.(b)에서의 3개의 특징점 a_1, a_2, a_3은 불필요한 조합이 된다. 반면에 b_1, b_2, b_3은 그들이 속해 있는 원을 나타내는데 중요한 하나의 후보점을 계산하는데 쓰인다. 따라서 먼저 각 패턴을 분류하여 각 패턴에 속해 있는 특징점들로부터만 조합추출에 의한 후보점 계산을 하면 보다 연산시간이 줄어들 것이다. 이와 같은 효과를 얻기 위해 여기에서는 먼저 (c)와 같이 낮은 해상도(예를 들어, 64×64)의 화상으로 변환한다. 여기에 4절과 5절에서 다룰 후보점 결정 방법과 매개변수공간의 동적분할 방법에 의한 후보

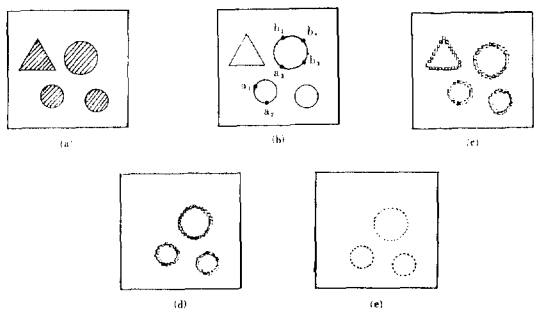


그림 1. 전반적인 절차
Fig. 1. Procedural Overview

점 누적 방법을 적용시켜 (d)와 같이 원하는 패턴을 개략적으로 구하여 각 패턴을 분류하여 기록해 둔다. 이렇게 분류된 패턴은 찾고자 하는 패턴일 가능성이 높은 것이다. 다음에는 분류된 각 패턴에 대한 원래의 해상도에서의 특징점들을 복원한다. 복원과정은 낮은 해상도에서의 화소와 원래의 해상도에서의 화소들간의 관계에 의해 쉽게 처리될 수 있다. 다음에는 분류된 각 패턴의 복원된 특징점들에 대해 다시 후보점 결정과 누적 방법을 이용하여 (e)와 같은 정확한 패턴들만을 확인추출한다.

3. Multigrid 표현법

화상을 컴퓨터에 표현해 놓는 방법으로서 본래의 화상의 해상도를 1/4씩 계속 줄여 가면서 여러 개의 낮은 해상도의 화상들을 만들어 모아 놓는 방법이 있다. 이것은 일종의 피라미드 구조처럼 보인다[1].

여기에서는 64×64의 해상도보다 높은 화상에 대해서 해상도를 낮추어 64×64의 해상도로 변환한 후, 이곳에 나타난 특징점들을 이용해 원하는 패턴들을 우선 개략적으로 분류하여 구하는데 목적이 있다.

해상도를 낮추는데 있어, 4개의 화소가 묶여 하나의 화소로 나타나게 되는데 이때 4개의 화소중에 2개 이상이 특징점이면 낮은 해상도의 화상에서도 특징점이 되게 한다. 이렇게 할 경우, 낮은 해상도 화상에 나타나는 특징점의 갯수는 바로 윗 단계 해상도 화상의 특징점의 갯수에 비해 반 이하로 줄어든다.

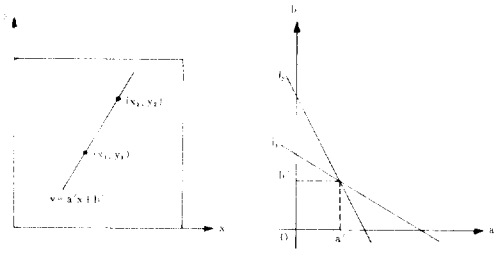
4. 후보점 결정

먼저 기존에 주로 사용해 오던 방법과 여기에서 사용하고자 하는 방법의 차이를 알아 보겠다. 이후로 전자의 방법을 일대다(one-to-many) 결정법이라고 하고 후자의 방법을 다대일(many-to-one) 결정법이라고 하겠다. 예를 들어, 직선(y=ax+b)을 찾는 경우를 생각해 보자.

일대다 결정법에서는 어느 한 특징점(x_1, y_1)을 직선식에 대입하여 다음과 같은 수식을 얻는다.

$$y_1=ax_1+b \qquad (1)$$

이 수식은 a와 b로 이루어지는 매개변수공간에서 그림 2. (b)에서와 같이 후보점들은 직선l_1과 같은 궤적으로 나타난다. 마찬가지로 특징점(x_2, x_2)는 다음과 같은 수식에 의해 직선l_2로 나타난다.



(a) 화상공간에서의 특징점들 (b) 매개변수공간에서의 후보점들

그림 2. 후보점과 매개변수공간
Fig. 2. Candidate points and Parameter Space

$$y_2=ax_2+b \qquad (2)$$

이러한 매개변수공간에서 실제 직선 y=a'x+b'를 나타내는 점은 (a', b')으로서 다른 점들은 불필요한 것들이다.

이에 비해, 다대일 결정법에서는 두 개의 특징점 (x_1, y_1), (x_2, y_2)에 의한 수식 (1), (2)로부터 다음과 같이 하나의 후보점 (a', b')만을 구해 매개변수공간에 누적시킨다.

$$a'=(y_1-y_2)/(x_1-x_2) \qquad (3)$$
$$b'=y_1-a'x_1 \qquad (4)$$

이러한 두 경우에 있어서, 누적되는 후보점의 갯수를 구해보자. 먼저 특징점의 갯수가 n개이고 매개변수의 갯수가 d개이며 각 매개변수가 q_i(i=1, , , , , d)개의 이산값을 가질수 있다고 하자. 그러면 일대다 결정법에서는 대략 $n\prod_{i=1}^{d-1}q_i$ 개의 후보점이 만들어지며 다대일 결정법에서는 $\binom{n}{2}$개가 만들어진다. 여기에서, 3절에 언급한 바에 따라 s단계만큼 해상도를 낮춤으로써 특징점의 갯수가 줄어든 경우를 생각해 보자. 즉 본래의 해상도는 $1/4^s$로 줄어든 경우이다. 이러한 경우에 있어서의 총 후보점의 갯수는 부록 A에서 알 수 있듯이 $\binom{n}{2}/4^{sd}\sim\binom{n}{2}/2^{sd}$개로 줄어든다. 이에 따라, 후보점 계산시간이 줄어들 수 있다. 그러나 낮춘 해상도에서의 처리이므로 개략적인 패턴분류효과만 얻을 수 있다. 따라서 이후에 복원된 특징점들로부터의 연산시간도 포함되어 비교해야 한다.

다음에는 후보점들의 집중성에 대해 알아 보겠다.

먼저 실제 입력된 화상에 존재하는 어떤 패턴을 이루고 있는 특징점의 갯수가 m이라 하자. 그러면, 이 패턴을 나타내는 매개변수값들로 이루어지는 매개변수공간상에서의 한 좌표점에 이상적으로 집중되어지는 후보점의 갯수는 일대다 결정법에서는 m개이지만 다대일 결정법에서는 $\binom{m}{n}$개가 된다. 이때 다대일 결정법에서는 잡음이나 서로 다른 패턴에 속해있는 특징점들로부터 구해지는 후보점들은 사방으로 흩어져 나타나므로 상대적으로 매우 높은 집중성을 나타낸다. 따라서 일대다 결정법에서와 같은 bias현상이나 intereference 현상[9]은 감소하게 된다. 아울러 이와같은 집중성으로 인해 5절에서의 매개변수공간의 동적분할 방법을 적용시키기에 보다 적합하며 과정도 간단해질 수 있다.

5. 매개변수공간의 동적분할

앞의 4절에서 구해진 후보점들은 매개변수공간에 누적된다. 여기에서 결국 얻고자 하는 것은, 확률적으로 다른 주변에 비해 많이 누적된 곳의 좌표값을 구함으로써, 원하는 패턴의 미지수값(매개변수값)을 결정하여 물체를 인식해 내려는 것이다. 즉, 매개변수공간에 누적된 값들로부터 지역적 최대값을 구하려는 것이다.

이를 위해, 기존의 Hough 변환법에서는 미리 매개변수공간을 고정된 간격으로 분할해 놓고 각 분할블럭(cell)에는 단지 자신의 블럭에 누적될 후보점들의 갯수만을 누적시키도록 하였다. 그런데, 이러한 경우에 매개변수가 증가함에 따라 매개변수공간을 위한 메모리 양이 지수적으로 급증하게 되어 컴퓨터의 메모리를 감당하기 어려운 상황에 도달하게 된다. 예를 들어, 3개의 매개변수를 각각 128개로 분할해 놓을 경우, 매개변수공간으로서 확보해야 할 배열의 크기는 128×128×128 개로서 컴퓨터에서는 4MB가 필요하게 된다.

이러한 급증현상은, 매개변수공간에서 실제로 많은 후보점들이 누적되는 부분만을 더욱 세밀하게 분할하여 각 분할블럭에 대해서만 하나의 메모리를 할당함으로써, 피할 수 있다[14),15)].

본 논문에서는 14)에서 제안된 동적분할 개념을 적용하되 4절에서의 다대일 결정법에 의해 구해진 후보점들을 누적시켰다. 이에 따라, 후보점들의 높은 집중성으로 인해 동적분할 개념이 보다 적합하게 응용될 수 있으며 아울러 보다 간단한 절차로서 동적분할을 행할 수 있다. 즉 15)에서와 같은 합병과정을 통해 구태여 각 분할블럭에 같은 갯수의 후보점들이 누적되도록 하지 않아도, 높은 집중성으로 인해 원하는 패턴에 해당되는 분할블럭은 더욱 세분되어 매우 작게 되므로 후에 지역적 최대값을 쉽게 찾을 수 있다. 또한 14)에서와 같이 각 블럭이 분할되는 위치를 새로운 후보점이 누적될 때마다 변화시킬 필요가 없다. 즉 분할트리에서 근노드(root node)로부터 그 후보점이 속하는 블럭노드까지의 경로(path)에 있는 블럭노드들의 분할 위치를 일일이 수정하지 않아도 된다. 이러한 잇점도 또한 후보점들의 높은 집중성으로부터 비롯된다.

다음에는 14)에서의 동적분할 개념을 예를 들어서 알아 보겠다. 이때 본 논문에서는 평균값에 의한 고정 분할 방법과 분할트리에 대한 간략화된 자료구조를 이용하여 좀 더 구체적으로 다루겠다.

예로서, 매개변수가 2개인 매개변수공간을 2차원 평면으로 나타낸 그림 3에서 알아보자. 맨 처음의 매개변수공간은 1개의 블럭으로 표현된다. 이곳에는 좌표값이 $x_0 \leq x < x_1$, $y_0 \leq y < y_1$의 범위에 있는 후보점들이 누적되어진다. 이곳에 i개의 후보점이 누적되어 평균좌표값이 $(x^{(i)}, y^{(i)})$일 때, 새로운 후보점 (x_c, y_c)가 누적되면, 새로운 평균좌표값 $(x^{(i+1)}, y^{(i+1)})$은 다음과 같이 구해진다.

$$x^{(i+1)} = \frac{ix^{(i)} + x_c}{i+1}$$

$$y^{(i+1)} = \frac{iy^{(i)} + y_c}{i+1}$$

이렇게 후보점들을 누적시키다가 누적갯수가 어느 분할임계치(threshold value)에 도달하면 그때까지 누적된 후보점들의 평균좌표값을 기준으로 4개의 부블럭으로 분할된다. 이것을 트리로 나타내면,

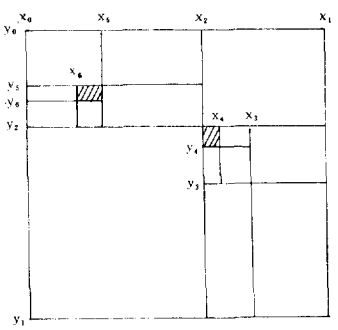


그림 3. 매개변수공간의 분할
Fig. 3. Spliting the parameter space

그림 4에서와 같이 처음의 블럭 A가 분할되어 4개의 분할블럭 B, C, D, E가 만들어진다. 이때 각 분할블럭들의 범위는 평균좌표값 (x_2, y_2)를 기준으로 하여 다음과 같이 된다.

$B : x_0 \leq x < x_2,\ y_0 \leq y < y_2$
$C : x_2 \leq x < x_1,\ y_0 \leq y < y_2$
$D : x_0 \leq x < x_2,\ y_2 \leq y < y_1$
$E : x_2 \leq x < x_1,\ y_2 \leq y < y_1$

이후로 누적될 후보점들은 4개의 분할블럭중에서 자신이 해당되는 분할블럭에 누적된다. 마찬가지로, 이러한 분할블럭중에서 후보점의 누적갯수가 분할임계치에 도달하는 것이 있으면 그 분할블럭은 다시 분할된다.

이렇게 필요할 때(분할임계치에 도달했을 때)에만 새로운 분할블럭들로 고정분할시켜 나감으로써 소요메모리의 급증을 피할 수 있다. 그리고 분할블럭이 너무 작을 경우에는 더 이상 분할하지 않고 단지 계속 누적되는 후보점들에 대해 평균좌표값과 누적된 후보점들에 대한 카운트값만 바꾼다.

위에서와 같이 구성된 분할트리를 dynamic quantization(DQ) 트리라고 하자. 이 DQ 트리를 컴퓨터에서 표현하면 그림 5와 같은 자료구조로 나타낼수 있다. 여기에서 알 수 있듯이, DQ 트리는 완전균형(full balanced) 트리[15]로서 평균점(average point)과 분할숫자(SPLIT)만 알면 따로 자식(children) 노드에 대한 포인터(pointer)가 필요없게 되어 이 트리에 대한 자료구조는 매우 간단해진다. 예를 들어, 분할블럭 O에 누적될 후보점은 먼저 근 노드 A 와 분할숫자 1을 보고 4개의 자식노드중에 분할블럭 B에 해당하는 것을 평균점으로부터 알게 되고 마찬가지로 분할블럭 B의 분할숫자 4와 평균점으로부터 분할블럭 H에 도달하게 되고 다시 분할블럭 O에 도달하게 된다.

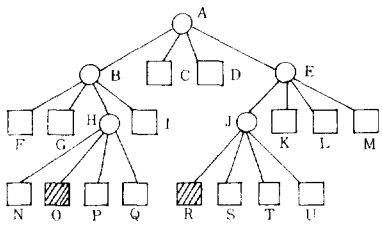


그림 4. DQ 트리
Fig. 4. DQ Tree

	NAME	Average x	point Y	CNT	SPLIT
start →					
1 →	A	x_2	y_2		1
	B	x_5	y_5		4
	C				
	D				
2 →	E	x_3	y_3		2
	J	x_4	y_4		3
	K				
	L				
3 →	M				
	R				
	S				
	T				
	U				
4 →	F				
	G				
	H	x_6	y_6		5
5 →	I				
	N				
	*O	a'	b'	TH+α	
	P				
	Q				

그림 5. DQ 트리의 자료구조
Fig. 5. Data Structure of DQ Tree

이렇게 구해진 매개변수공간으로부터 지역적 최대값을 구하는 것은 간단하다. 매개변수 공간에서 실제의 패턴을 나타내는 분할블럭으로서는 크기가 더 이상 분할되지 않을 정도로 작으면서 그곳에 누적된 후보점의 갯수가 분할임계치 이상이어야 한다. 그런데 위에서와 같은 누적 과정을 거치면 분할임계치 이상의 후보점들이 누적되어 있는 분할블럭은 필연적으로 매우 작게 되어 있다. 따라서 그림 4와 같은 DQ트리에서 각 단말노드(terminal node) 블럭의 크기를 알기 위해 근 노드로부터 경로를 따라 일일이 탐색해 내려갈 필요없이, 단순히 그림 5의 자료구조에서 앞에서부터 순차적으로 각 분할블럭의 카운트(CNT)값이 분할임계치 이상인가를 조사하면 된다. 만약 이러한 분할블럭이 존재하면, 그것의 평균점의 좌표값이 실제의 매개변수값이 된다. 예를 들어, 탐색과정에서 분할블럭 O가 분할임계치(=TH)보다 많은 카운트값을 갖고 있다면, 평균점의 좌표값 a', b'가 찾고자 하는 패턴의 매개변수값이 된다.

6. 분석 및 실험 결과

여기에서는 먼저 후보점을 결정하기 위한 일대다

결정법과 다대일 결정법에 있어서의 연산시간을 알아보고, 아울러 multgrid 표현법에 의한 단계적 분석과정에서의 연산시간도 구해 보겠다. 또한 매개변수공간을 누적 배열로 구현하는 방법과 동적 분할에 의한 DQ트리 표현법에 있어서의 소요메모리 양에 대해서 알아 보겠다. 다음에는 IBM PC/AT에서의 시뮬레이션 결과를 소개하겠다.

6. 1 분석

먼저 다음과 같은 상수들을 정의하자.

N^2 : 입력 화상의 해상도
n : 특징점의 갯수
d : 매개변수의 갯수
p : 입력 화상에 존재하는, 찾고자 하는 패턴의 갯수
q_i : 각 매개변수가 취할수 있는 이산값의 갯수 (i=1...d)
k : 분할임계치
n_j : q개의 각 패턴을 이루는 특징점의 갯수(j=1...p)
s : 낮은 해상도로 줄인 정도
(예, 256×256에서 64×64이면 s=2)

6. 1. 1 연산시간 산출

일반적으로 동적 분할방법이 누적배열 사용방법보다 많은 누적시간이 소요됨을 추측할 수 있다. 그러나 본 논문에서의 지역적 최대값을 구하는 과정은 누적배열에서의 지역적 최대값을 구하는 과정보다 시간이 덜 소요됨을 쉽게 알 수 있다. 여기에서는 후보점을 누적시키는 시간이나 지역적 최대값을 구하는 시간에 대한 비교는 하지 않고 단지 후보점 결정에 소요되는 시간에 대해서만 알아 보겠다.

먼저 일대다 결정법에서는 n개의 특징점에 대해서 (d−1)개의 각 매개변수가 취할 수 있는 이산값의 조합 각각에 대해 하나의 수식으로부터 하나의 후보점이 결정되므로 연산시간은 C1이라 할 수 있다. 다대일 결정법에 있어서는 n개의 특징점으로부터 d개씩을 조합 추출한 모든 경우의 수에 대하여 d개의 수식으로부터 한 개의 후보점이 결정된다. 따라서 연산 시간은 C2와 같다. 단, 하나의 수식을 계산하는 시간은 같다고 가정한다. 이에 비해, s단계만큼 낮추어진 해상도에서의 특징점들($n/4^s \sim n/2^s$개)로부터 d개 씩을 조합 추출하여 d개의 수식에 적용하여 한개의 후보점을 결정하는데는 C3에서의 첫 항에 해당하는 연산횟수가 필요하다. 또한 p개의 패턴에 대해 다시 다대일 결정법에 의한 후보점을 계산해내는데 필요한 연산횟수는 C3에서의 둘째 항과 같다. 따라서 본 논문에서 사용한 후보점 결정에서 필요로 하는 연산 시간은 C3가 된다.

$$C1=n\prod_{i=1}^{d-1}q_i$$

$$C2=b\binom{n}{d}$$

$$C3=d\binom{nm^s}{d}+d\sum_{i=1}^{p}\binom{n_i}{d},\ 2<m<4$$

여기에서 $q_i=q$, $n_i=\frac{n}{ap}$, $i=1,\cdots,d$, $a<1$ 이라고 하자. 또한 $M=n/q$라 하자. 그러면 위의 세 수식은 다음과 같이 표현될 수 있다.

$$C11=q^dM$$

$$C22=d\ q^dM^d$$

$$C33=dq^d\left(\frac{1}{m^{sd}}+\frac{1}{a^dp^{d-1}}\right)M$$

여기에서 C11와 C22의 교점 M_1, C11와 C33의 교점 M_2^x는 다음과 같이 구해진다.

$$M_1=\frac{1}{d}$$

$$M_2=\frac{m^{sd}a^dp^{d-1}}{d(a^dp^{d-1}+m^{sd})}$$

따라서 C22와 C33는 특징점의 갯수가 각각 $M_1 q$, $M_2 q$ 보다 작은 화상에서는 C11에 비해 작게 되어 적용시키기에 합당하나 갯수가 그 이상이 되면 값이 지수적으로 급증하여 부적당하다.

여기에서 m, a에 대하여 각각 임의의 적당한 값 3, 2를 대입시켜, d, s, p값의 변화에 따른 M_1, M_2 값의 변화를 표1에 나타내었다. 이 표에서 알수 있듯이, 본 논문에서 제안된 후보점 결정법은 d, s, q 값의 증가에 따라 보다 복잡한(특징점의 갯수가 많은) 화상에 대해서도 적용시킬 수 있다.

6. 1. 2 소요 메모리 산출

매개변수공간으로서 누적배열을 이용한 경우에 소요되는 메모리는 화상저장용 배열을 위한 메모리 N^2과 더해져 S1으로 표기될 수 있다. 동적분할 방법에서는 DQ트리용으로서 (분할에 따른 파생 노드수)x(한 노드를 위한 메모리 양)x(총 분할 횟수)개의 메모리가 필요하다 다. 여기에서 총 분할횟수는 모든 후보점들이 한 곳에 집중되는 경우에 최대값을 가지며 「(총후보의 갯수)/(분할 임계치)」로 서 표현될 수 있다. 그러나 실제에 있어서는 후보점들

이 여러 노드에 분포될 뿐만 아니라 분할된 블럭이 매우 작은 경우에는 더 이상 분할을 하지 않으므로 총 분할횟수는 최대값보다 훨씬 작은 값으로 나타난다(실제 실험에서 약 0.02배로 나타났음). 본 논문에서는 낮추어진 해상도의 화상을 저장하기 위한 배열로서 $N^2/4^s$개의 메모리가 더 필요하게 되어 전체 소요메모리는 S2로 나타낸다.

$$S1=\prod_{i=1}^{d} q_i+N^2$$

$$S2=N^2+N^2/4^s+2^d(d+2)\alpha\lceil(n/m_d^s)/k\rceil,$$
$$\alpha\ll 1 \text{ and } 2\leq m\leq 4$$

여기에서 S1과 S2의 직접 비교는 곤란하나 실제 실험이나 논리상의 추론으로부터 S2가 S1에 비해 매우 작음을 알 수 있다.

6. 2실험 결과

여기에서는 매개변수공간의 분할된 모양을 알아보고 또 기존의 변환법과의 연산시간및 소요메모리 양을 비교하기 위하여, 그림 6와 같은 128×128해상도의 화상에 sobel 연산자를 적용시킨 후 그림 7과 같이 직선을 찾아보았다.

이 실험은 IBM PC/AT에서 시뮬레이트되었다.

이 실험에서 소요된 메모리와 시간은 표 2에 정리하였다. 여기에서 누적배열 사용시에 매개변수인 직선의 기울기(a)와 절편(b)의 허용범위와 이산값에 갯수는 사용된 컴퓨터의 메모리 제한으로 인해 적절히 제한시킨 것이다. 따라서 매개변수공간은 32KB로 표현되었다. 이에 비해 동적분할에 의한 표현법에서는 매개변수로서 부동점(floating point)형

표 1. M_1과 M_2값의 변화표

Table 1. Values for M_1 and M_2 according to variations of d, p, and s.

d	p	M_1	M_2	
			S=1	S=2
2	1	0.5	1.38	1.91
2	2	0.5	2.12	3.64
2	3	0.5	2.57	5.23
3	1	0.33	2.06	2.64
3	2	0.33	4.88	10.22
3	3	0.33	6.55	21.84
4	1	0.25	3.34	3.99
4	2	0.25	12.40	31.39
4	3	0.25	17.05	101.33

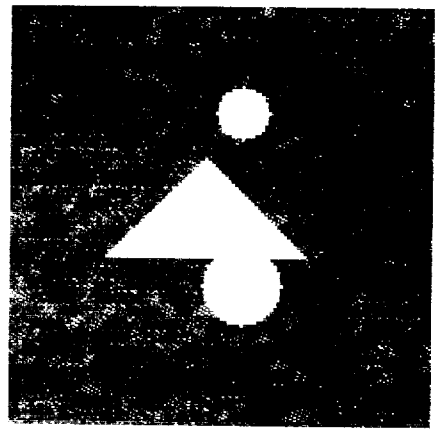


그림 6. 입력된 다치화상

Fig. 6. Inputed Grey Image

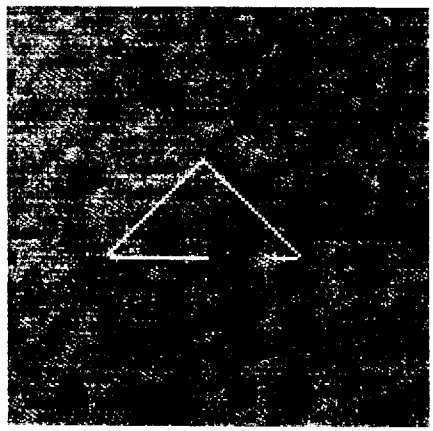


그림 7. 추출된 패턴(직선)

Fig. 7. Extracted Lines

의 변수를 사용하였으므로 허용오차는 대략 10^{-6}정도이며 허용범위는 임의로 설정 가능하다. 여기에서의 소요메모리는 입력화상에 따라 따라 좌우된다. 이 실험에서는 4.7KB가 소요되어 누적배열 표현법보다 훨씬 적음을 알 수 있다. 이때 매개변수공간의 분할은 그림 8과 같다.

누적배열 사용시의 연산시간은 매개변수들의 이산값의 갯수에 따라 좌우되고 동적분할법에서는 특징점의 갯수에 따라 좌우되므로 서로 비교하기 곤란하다. 이 실험에서는 160개의 특징점과 128개의 이산값을 사용했을 때 표 2에서와 같은 연산시간을 얻었다.

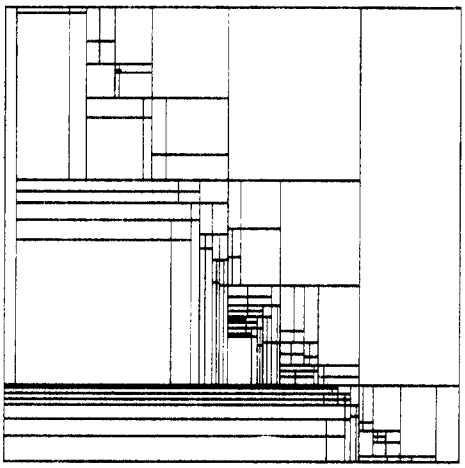


그림 8. 분할된 매개변수공간
Fig. 8. Splitted Parameter Space

표 2. 소요메모리와 소요시간의 비교표
Table 2. Comparison of the consumed memory and time between conventional transform and the proposed transform in this paper.

		일대다 결정법, 누적배열	다대일 결정법, 등적분할
소요메모리	매개변수공간용	$-32 \leqq a < 32$ $-64 \leqq b < 192$ $\triangle a = 0.5$, $\triangle b = 1.0$ 32KB 소요	$-32 \leqq a < 32$ $-1000 \leqq b < 1000$ $\triangle a = \triangle b = 10^{-6}$ 437개의 노드 사용 4.7KB 소요 (=437×11 Byte)
	화상 저장용	16KB	20KB
	전 체	48KB	24.7KB
소요시간		37.35 초	28.16 초

7. 결 론

본 논문에서는 먼저 소요메모리의 급증현상을 매개변수공간의 동적분할 방법에 의해 해결할 수 있었다. 이때, 다대일 결정법에 의한 후보점들을 매개변수공간에 누적되도록 함으로써 후보점들의 높은 집중성에 의해 분할절차및 지역적 최대값 결정과정이 매우 간단해졌다. 또한 다대일 결정법에서 단계적 분석추출방법을 사용함으로써 특징점의 갯수 증가에 따른 연산시간의 급증현상을 완화시킬 수 있었다.

본 논문에서의 변환방법은 매개변수공간의 차원이 높아질수록 소요메모리의 감소효과가 커지며 보다 복잡한 화상에 대해서도 적용가능해짐을 알 수 있다.

앞으로, 다대일 결정법에 의한 후보점들의 높은 집중성으로 인해 부수효과(side effect)의 감소가 기대되는 바, 이에 대한 자세한 분석과 동적분할임계치의 효율적인 결정방법에 대한 연구를 할 것이다. 아울러 함수패턴이 아닌 임의의 패턴에 대해서도 본 변환법을 적용가능하도록 확장에 관한 연구도 진행할 것이다.

부록 A. 낮춘 해상도 화상에서의 총 후보점 갯수의 계산

본문의 3절에서 언급한 바에 따라 해상도를 한 단계씩 줄일 때마다 특징점의 갯수는 1/4-1/2개로 줄어든다. 따라서 s단계만큼 줄인 경우에는 $1/4^s$-$1/2^s$개로 줄어든다. 그러면 원래의 화상에서의 특징점의 갯수가 n개인 경우에 낮춘 해상도 화상에서의 총 후보점 갯수는 $\binom{n/4^s}{d} \sim \binom{n/2^s}{d}$개가 된다. 여기에서 d는 매개변수의 갯수이다. 또한 $\binom{n/k}{d}$는 아래와 같이 표현될 수 있다.

$$\binom{n/k}{d} = \frac{(n/k)!}{(n/k-d)!\,d!}$$

$$= \frac{n!}{(n-d)!\,d!} \cdot \frac{(n-d)!\,d!}{n!} \cdot \frac{(n/k)!}{(n/k-d)!\,d!}$$

$$= \binom{n}{d} \cdot \frac{(n/k)\,(n/k-1)\,\cdots\,(n/k-d+1)}{n(n-1)\,\cdots\,(n-d+1)}$$

$$= \binom{n}{d} \cdot \frac{1}{k^d} \cdot \frac{n(n-k\cdot 1)\,\cdots\,(n-k\cdot(d-1))}{n(n-1)\,\cdots\,(n-(d-1))}$$

$$< \binom{n}{d}/k^d, \quad k > 1$$

여기에서 k는 4^s또는 2^s이므로 총 후보점의 갯수는 다음과 같이 구해진다.

$$\binom{n}{d}4^{sd} \sim \binom{n}{d}/2^{sd}$$

REFERENCE

1) D. H. Ballard and C. M. Brown, Computer Vision, Englewood Cliffs, NJ:Prentice-Hall, 1982.
2) P. V. C. Hough, "Method and means for re-

cognizing complex patterns," U.S.Patent 3069654, 1962.

3) F. O. Gorman and M. B. Clowes, "Finding picture edges through collinearity of feature points," IJCAI, 1973, pp. 543-555.

4) C. Kimme, D. Ballard, and J. Sklansky,"Finding circles by an array of accumulators,"CACM 18, 2, 1975. pp. 120-122.

5) H. Wechsler and J. Sklansky, "Finding the rib cage in chest radiographs," Pattern Recognition 9, 1977, pp. 21-30.

6) R. O. Duda and P. R. Hart, "Use of the Hough transformation to detect lines and in curves pictures," CACM 15, 1, 1972, pp.11-15.

7) D. H. Ballard, "Parameter networks: Towards a theory of low-level vision," IJCAI, 1981, pp. 1068-1078.

8) C. M. Brown, M. B. Curtiss, and D. B. Sher, "Advanced Hough Transform Implementations," IJCAI, 1081-1085.

9) C. M. Brown, "Inherent Bias and Noise in the Hough Transform," IEEE PAMI, Vol. 5, 1983, pp. 493-505.

10) S. D. Shapiro, "Transformations for the computer detection of curves in noisy pictures," CGIP, Vol. 4, 1975, pp. 328-338.

11) S. D. Shapiro, "Properties of transforms for the detection of curves in noisy pictures," CGIP, Vol. 8, 1978, pp. 219-236.

12) S. D. Shapiro and A. Iannino, "Geometric constructions for predicting Hough transform performance," IEEE PAMI, Vol. 1, 1979.

13) J. Sklansky, "On the Hough technique for curve detection," IEEE Trans. Comput., Vol. C-27, Oct. 1978, pp.923-926.

14) K. R. Sloan, "Dynamically Quantized Pyramids," IJCAI, 1981, pp. 734-736.

15) J. O'Rourke, "Dynamically Quantized Spaces for Focusing the Hough Transform," IJCAI, 1981, pp. 737-739.

16) D. H. Ballard, "Generalizing the Hough transform to detect arbitrary shapes," Pattern Recognition, Vol. 13, No. 2, 1981, pp. 114-115.