추적 조건 분석에 의한 개선된 외곽선 추적 기법들

정철호[O] 서종훈[1] 한탁돈[2]
연세대학교 컴퓨터과학과
{bright[O], hantack[2]}@kurene.yonsei.ac.kr, flamme4u[1]@yonsei.ac.kr

Advanced Contour Tracing Algorithms based on Analysis of Tracing Conditions

Cheolho Cheng[O] Jong-Hoon Seo[1] Tack-Don Han[2]
Dept. of Computer Science, Yonsei University

요 약

외곽선 추적 알고리즘은 영상 인식 및 표현에 있어서 물체의 기본 성질을 파악하는데 중요하다. 따라서 많은 알고리즘들이 연구되어 왔으며, 이중에는 간단한 경계선 추적자 알고리즘(SBF: Simple Boundary Follower)이다. 이외에도 수정된 간단한 경계선 추적자 알고리즘 (MSBF: Modified Simple Boundary Follower), 개선된 간단한 경계선 추적자 알고리즘(ISBF: Improved Simple Boundary Follower), 무어-네이버 추적 알고리즘(MNT: Moore-Neighbor Tracer), 방사형 탐색 알고리즘(RSA: Radial Sweep Algorithm), 그리고 Theo Pavlidis 알고리즘(TPA)이 있다. 이러한 알고리즘들은 추적 경로 특성들이 다르며 각기 장점과 제약성이 있다. 외곽선 알고리즘들의 제약성은 크게 두 가지로 나눌 수 있다. 하나는 알고리즘 특성에 따라 외곽선 픽셀간 인접 형태에 따라 추적하지 못하는 경우가 존재할 수 있다는 것이다. 또 다른 하나는 외곽선 추적 알고리즘의 시작과 종료 조건에 따라서 특정 위치 픽셀들을 찾지 못하는 경우도 존재한다는 점이다. 본 논문에서는 이러한 문제점들을 중심으로 외곽선 추적 알고리즘들의 성능을 분석하였다. 또한, ISBF의 시작 조건과 TPA의 인너코너 추적을 개선하는 기법들을 제안하여 이를 해결토록 하였다. 실험 결과 제안한 기법들은 외곽선 추적 성능을 개선하는데 효과적이었다.

1. 서 론

외곽선 추적 알고리즘(contour tracing algorithm)은 경계선 추적 알고리즘(boundary following algorithm)[1]이라고도 하며, 영상에서 특정한 물체의 외곽선을 추적하는 알고리즘이다. 영상에서 물체의 외곽선을 추적하는 것은 다양한 용도로 사용된다[2]. 물체를 배경 이미지로부터 분리할 수 있으며, 물체 외곽선 길이를 알 수 있으므로 물체의 크기와 모양을 추정할 수 있다. 또한 외곽선을 구성하는 픽셀들의 방향 정보에 따라 물체의 선분 정보와 방향 변화 정보를 알 수 있으므로 물체의 특징점을 찾아내는 경우에도 매우 유용하며, 물체를 영상에 표현하거나 묘사할 경우에도 이용된다[3],[4]. 근래에는 컬러나 회색조 영상에서 외곽선을 추적하는 기법[2], 물체의 내부까지 추적하는 기법[5]등 다양한 종류의 외곽선 추적 알고리즘이 있으나, 기본적으로는 이진화를 통해 생성된 이진 이미지(binary image)의 물체를 인식하는 방식을 확장한 것이라고 할 수 있다.

외곽선 추적 알고리즘으로는 간단한 경계선 추적자 알고리즘(SBF: Simple Boundary Follower)[3],[6],[7]과 이의 변형 알고리즘인 수정된 간단한 경계선 추적자 알고리즘 (MSBF: Modified Simple Boundary Follower)[4], 개선된 간단한 경계선 추적자 알고리즘(ISBF: Improved Simple Boundary Follower)[8], 무어-네이버 추적 알고리즘(MNT: Moore-Neighbor Tracer)[9], 방사형 탐색 알고리즘(Radial Sweep Algorithm)[9],[10], 그리고 Theo Pavlidis 알고리즘[9],[11]이 있다. 이 알고리즘들은 알고리즘이 매우 간단하므로 적용이 용이한 장점이 있다. 그러나 이들 알고리즘이 모든 경우에 적합한 것은 아니며, 각자 알고리즘들이 유효하지 않은 경우들이 있다.

추적자 알고리즘은 기본적으로 외곽선상의 한 점에서 시작하여 물체의 가장 바깥쪽에 위치한 외곽선들을 추적한 후, 최초 출발지로 돌아오면 종료된다. 그러나 각각의 추적자 알고리즘들은 이러한 조건을 만족하지 못하는 경우가 있는데, 주로 두 가지 요인에 기인한다.

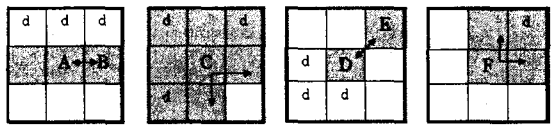
첫 번째 요인은 외곽선상의 현재 추적자가 위치한 픽셀과 다음에 추적할 외곽선 픽셀과의 상대적 위치로 인해 추적할 수 있는 경우와 그렇지 않은 경우가 존재한다는 것이다. 두 번째 요인은 시작과 종료 조건에 따라 추적이 어려운 경우가 발생한다는 것이다. 한 예로 어떤 알고리즘의 종료조건이 단순히 추적자가 다시 시작위치로 돌아올 때 추적을 종료하는 것이라 하자. 이 경우 외곽선의 중간에서 출발한 추적자가 한 쪽 외곽선만 추적했다가 출발했던 곳으로 돌아오면 추적을 종료하게 되므로, 아직 추적하지 못한 외곽선이 발생하게 된다.

본 논문에서는 앞에서 언급한 외곽선 추적 알고리즘들의 비일관성 문제와 시작과 종료 조건 문제를 중심으로 각기 알고리즘들의 장단점을 분석하였다. 또한 이러한 문제들을 일부, 혹은 모두를 해결할 수 있는 기법과 실험결과를 제시하였다.

2. 관련 연구

외곽선 추적 알고리즘의 추적경로는 크게 직선과 코너 방향으로 나눌 수 있다. 직선 경로는 현재 추적자가 위치한 외곽선 픽셀에서 인접한 외곽선 픽셀이 상하 혹은 좌우로 직선으로 연결된 경우이며, 코너연결은 경로가 코너를 형성하는 경우이다. 코너의 종류는 인너코너(inner

corner), 아우터 코너(outer corner), 그리고 인너-아우터 코너(inner-outer corner)로 나눌 수 있다. 그림 1(b)에서 인너 코너 C의 경우 상하좌우의 4방향은 반드시 흑색이어야 하지만, 대각선 방향들은 흰색이어도 된다. 한편, 그림 1(d)에서 d 픽셀이 흰색이면, F가 인너코너처럼 보이지만, 인너코너 조건을 만족하지 않는다. 기존의 추적알고리즘들을 살펴보면 다음과 같다.



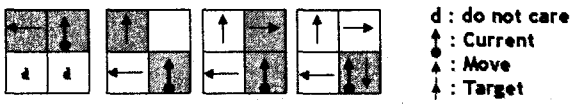
(a)직선의 예 (b)인너코너 (c)인너-아우터코너 (d)아우터코너

그림 1 연결관계 (d: do not care)

2.1 간단한 경계선 추적자 알고리즘

간단한 경계선 추적자 알고리즘(SBF: Simple Boundary Follower)은 Papert의 거북이 알고리즘(turtle algorithm)[6], 혹은 직각 추적 알고리즘(square tracing algorithm)[9]으로 알려져 있으며 가장 간단한 외곽선 추적 알고리즘이다. 외곽선 추적 기법을 설명하기 위해 편의상 관심 영역, 즉 외곽선을 흑색, 배경 이미지를 흰색이라고 가정하자. 그리고 현재 픽셀의 위치를 P, 방향정보를 d라고 하고, 추적자를 T(P,d), 추적 시작 정보를 S(P,d)라고 하자, 그리고 P_a를 d방향성을 가진 P의 a방향 위치, d_a를 d방향을 기준으로 a방향을 나타낸다고 하면, SBF의 프로시저는 다음과 같다.

① T(P,d) ← S(P,d)
② Do
③ if P = black then T(P,d)←T(P_{left},d_{left})
④ else T(P,d)←T(P_{right},d_{right})
⑤ While T(P,d) ≠ S(P,d)

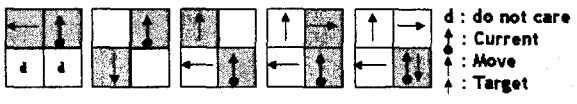


(a)좌측인접 (b)좌상측 인너-아우터코너 (c)상측인접픽셀 (d)아우터 코너

그림 2 SBF에서 처리가능한 외곽선 픽셀 인접형태들[8]

2.2 수정된 경계선 추적자 알고리즘

수정된 경계선 추적자 알고리즘(MSBF: Modified Simple Boundary Follower)은 SBF의 비일관성 중의 하나인 추적자의 좌하에 위치한 인너-아우터 픽셀의 위치를 탐색하기 위한 알고리즘이다. 현재 추적자의 위치와 추적자의 좌하 방향이 흑색, 좌와 하는 흰색일 경우, 추적자는 좌하 픽셀로 이동하며, 방향성은 반대로 가진다[4]. MSBF가 추적할 수 있는 경우는 그림 3과 같고, 프로시저는 다음과 같다.



(a)좌측 픽셀 (b)좌하측 인너-아우터 코너 (c)좌상측 인너-아우터 코너 (d)상측의 픽셀 (e)아우터 코너

그림 3 MSBF에서 처리가능한 외곽선 픽셀 인접형태들[8]

① T(P,d) ← S(P,d)
② Do
③ if P = black then
④ if tag=0 and $P_{left\text{-}rear}$=black and P_{left}=white and P_{rear}=white
⑤ then T(P,d)←T($P_{left\text{-}rear}$,d_{rear}) and tag←1
⑥ else T(P,d)←T(P_{left},d_{left}) and tag←0
⑦ else T(P,d)←T(P_{right},d_{right})
⑧ While T(P,d) ≠ S(P,d)

2.3 개선된 경계선 추적자 알고리즘

개선된 경계선 추적자 알고리즘(ISBF: Improved Simple Boundary Follower)[8]는 SBF와 MSBF의 문제점인 상단 픽셀이 인너코너인 경우를 처리하기 위해 변형된 알고리즘으로 본 논문의 선행연구이다. SBF와 MSBF와 달리 흰색 픽셀에서 이동연산이 없는 것이 특징이며 외곽선 픽셀들의 추적경로를 6가지로 분류하여 처리한다.

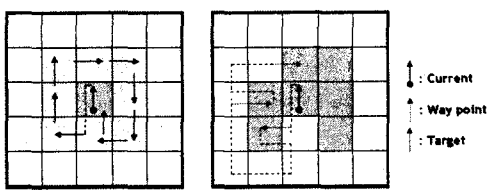
① T(P,d) ← S(P,d) and tag←0 where P is on black
② Do
③ if P_{left}=black then T(P,d)←T(P_{left},d_{left})
④ else if tag=0 and $P_{left\text{-}rear}$=black and P_{rear}=white
⑤ then T(P,d)←T($P_{left\text{-}rear}$,d_{rear}) and tag←1
⑥ else
⑦ tag←0
⑧ if $P_{left\text{-}front}$=black
⑨ if P_{front}=black then
⑩ T(P,d)←T(P_{front},d) and T(P,d)←T(P_{left},d)
⑪ else T(P,d)←T($P_{left\text{-}front}$,d)
⑫ else if P_{front}=black then T(P,d)←T(P_{front},d_{right})
⑬ else T(P,d)←T(P,d_{rear})
⑭ While T(P,d) ≠ S(P,d)

(a) 좌측 픽셀 (b) 좌하측 인너-아우터 코너 (c) 좌상측 인너-아우터 코너 (d) 상측의 인너코너 (e) 상측의 픽셀 (f) 아우터 코너

그림 4 물체 경계선 픽셀의 배치 및 경계선 픽셀간 추적 경로[8]

2.4 무어-네이버 추적 알고리즘

무어-네이버 추적 알고리즘(MNT: Moore-Neighbor Tracer)은 8-방향의 연결성을 추적할 수 있도록 고안된 알고리즘이다[9]. 그림 5와 같이 현재 추적자의 위치에서 좌하, 좌, 좌상, 상, 우상, 우, 우하, 하의 순서로 탐색하며 화살표와 같은 방향성을 가진다. MNT의 프로시저는 다음과 같다.



(a) 추적자의 추적 순서 (b) 추적자의 추적 경로

그림 5 MNT의 외곽선 추적 경로

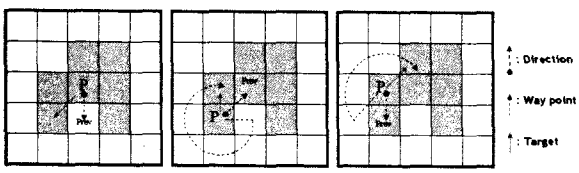
① T(P,d) ← S(P,d)
② count_visit_start = 0;
③ Do
④ if $P_{left\text{-}rear}$=black then T(P,d)←T($P_{left\text{-}rear}$,d_{left})
⑤ else if P_{left}=black then T(P,d)←T(P_{left},d)
⑥ else if $P_{left\text{-}front}$=black then T(P,d)←T($P_{left\text{-}front}$,d)
⑦ else if P_{front}=black then T(P,d)←T(P_{front},d_{right})
⑧ else if $P_{right\text{-}front}$=black then T(P,d)←T($P_{right\text{-}front}$,d_{right})
⑨ else if P_{right}=black then T(P,d)←T(P_{right},d_{rear})
⑩ else if $P_{right\text{-}rear}$=black then T(P,d)←T($P_{right\text{-}rear}$,d_{rear})
⑪ else if P_{rear}=black then T(P,d)←T(P_{rear},d_{left})
⑫ else END // isolated
⑬ if T(P) = S(P) then count_visit_start←count_visit_start+1
⑭ While T(P,d) ≠ S(P,d) or count_visit_start < n

2.5 방사형 탐색 알고리즘

방사형 탐색 알고리즘(RSA: Radial Sweep Algorithm)[10]은 MNT와 거의 동일한 알고리즘이다[9]. RSA는 그림 6과 같이 먼저 현재 추적자 위치(P)에서 선행 추적자 위치(prev)간 방향성을 구한 후 이 방향 선분의 우측에 있는 픽셀부터 탐색하여 흑색 픽셀이 나올 때까지 추적한다.

RSA의 추적자가 가진 방향정보는 선행한 외곽선 픽셀 위치와 관계에 의해 항상 새로이 설정되므로, 종료 조건에 사용할 수 없다. 따라서 추적자가 이미 지났던 경로를 다시 지날 경우 추적을 종료한다. 이전 추적자의 위치를 P_{prev}라고 하면 이의 프로시저는 다음과 같다.

① B ← Null // B is a contour list
② T(P) ← S(P) and P_{prev}← neighbor white pixel of P
③ While
④ if path of (P_{prev},P) exists in B then STOP
⑤ d← rection from P to P_{prev}
⑥ else append P to B
⑦ if $P_{right\text{-}front}$=black then T(P)← ($P_{right\text{-}front}$)
⑧ else if P_{right}=black then T(P)← (P_{right})
⑨ else if $P_{right\text{-}rear}$=black then T(P)← ($P_{right\text{-}rear}$)
⑩ else if P_{rear}=black then T(P)← (P_{rear})
⑪ else if $P_{left\text{-}rear}$=black then T(P,d)← ($P_{left\text{-}rear}$)
⑫ else if P_{left}=black then T(P)← (P_{left})
⑬ else if $P_{left\text{-}front}$=black then T(P)← ($P_{left\text{-}front}$)
⑭ else if P_{front}=black then T(P)← (P_{front})
⑮ else STOP // isolated
⑯ End While

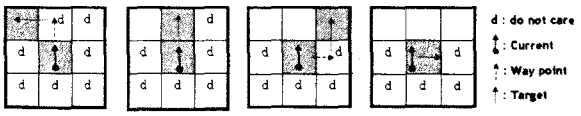


(a) RSA시작 경로 (b) 두 번째 추적경로 (c) 세 번째 추적 경로
그림 6 RSA의 추적과정

2.6 Theo Pavlidis 알고리즘

Theo Pavlidis 알고리즘(TPA)[11]은 현재 추적자의 위치로부터 좌상, 상, 우상 방향에 위치한 외곽선 픽셀을 목표로 추적해나간다. 해당 방향에 외곽선 픽셀이 없으면 우측으로 90도 회전한다. 시작시 추적자의 왼쪽 픽셀은 반드시 흰색이어야 한다[10]. TPA는 추적자에게 방향성이 있으나 종료 조건으로 사용할 수는 없으며, 시작 위치에 n번 이상 진입할 때 종료한다. TPA의 프로시저는 다음과 같다.

① T(P,d) ← S(P,d), where P_{left} = white
② count_visit_start = 0;
③ Do
④ count_rotate ← 0
⑤ Do
⑥ if $P_{left\text{-}front}$=black then T(P,d)←T($P_{left\text{-}front}$,d_{left})
⑦ else if P_{front}=black then T(P,d)←T(P_{front},d)
⑧ else if $P_{right\text{-}front}$=black then T(P,d)←T($P_{right\text{-}front}$,d)
⑨ else T(P,d)←T(P,d_{right}) and count_rotate←count_rotate +1
⑩ While count_rotate <= 3
⑪ if T(P) = S(P) then count_visit_start←count_visit_start+1
⑫ While count_visit_start < n



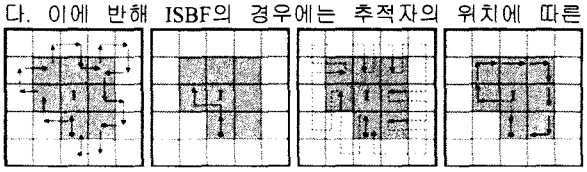
(a)좌상픽셀 추적 (b)상단픽셀 추적 (c)우상픽셀 추적 (d)이외의 경우
그림 7 TPA의 추적 순서

3. 외곽선 추적 알고리즘의 추적불가능과 종료조건 분석

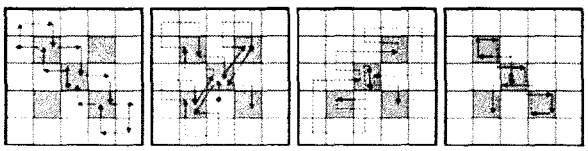
3.1 추적 불가능 조건 분석

외곽선이 상하 선분이나 좌하 선분인 경우 모든 외곽선 추적 알고리즘들은 이를 추적할 수 있다. 그러나 코너인 경우, 특히 인너코너와 인너-아우터 코너인 경우는 추적자의 조건에 따라 감지할 수도 있고, 그렇지 않을 수도 있다. 한 예로 SBF는 그림 8(a)와 같이, 상측 픽셀 I가 인너코너인 경우 이를 감지하지 못한다. 그러나 추적자의 좌측 픽셀이 인너코너라면, 이 경우에는 추적할 수 있다. 이렇듯 추적자의 조건에 따라 코너를 인식할 수도 있고, 그렇지 않을 수도 있는 경우를 비일관성이라고 한다. SBF는 상측픽셀이 인너코너인 경우와 좌하측에 인너-아우터 코너인 경우에 추적을 못한다.

MSBF는 좌하측 인너-아우터 코너의 비일관성을 해결한 반면, 인너코너의 비일관성은 여전히 가진다. MNT, RSA, TPA의 경우에는 인너코너 픽셀은 추적하지 못한다. 이에 반해 ISBF의 경우에는 추적자의 위치에 따른



(a) SBF와 MSBF (b) ISBF (c) MNT와 RSA (d) TPA
동그라미 화살표: 추적자, I: 인너코너, ↑ 이동, 점선화살표: 탐색경로
그림 8 인너코너 처리



(a) SBF (b) MSBF와 ISBF (c) MNT와 RSA (d) TPA
그림 9 인너-아우터 코너 처리

표 1 알고리즘별 연결성 추적 비교
(0: 추적가능, V: 비일관성 있음, X: 추적불가능)

	직선연결	인너코너	인너-아우터	아우터코너
SBF	0	V	X	0
MSBF	0	V	0	0
ISBF	0	0	0	0
MNT	0	X	0	0
RSA	0	X	0	0
TPA	0	X	0	0

비일관성이 없다. 그림 8은 알고리즘의 인너코너 처리를 보여주고 있으며, 그림 9는 인너-아우터 코너의 처리 경로를 보여준다. 인너-아우터 코너의 종류는 모두 네 가지이지만 실제로는 두 가지 종류로서 처리된다. 즉, SBF, MSBF, ISBF, MNT, RSA의 경우에는 우상과 우하의 인너-아우터 코너들이 추적자의 회전을 통해 좌상 혹은 좌하의 인너-아우터 코너로서 인식된다. 이에 반해 TPA는 네 가지가 좌상과 우상 인너-아우터 코너로서 처리된다. 표 1은 연결유형에 따른 알고리즘별 추적 가능성을 보여준다.

3.2 시작 및 종료 조건에 따른 성능 분석

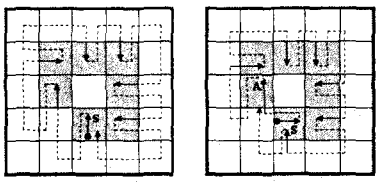
외곽선 추적자 알고리즘들의 시작과 종료 조건에 따른 성능 분석을 위해 추적자의 시작 위치를 흰색 픽셀에서 흑색 픽셀로 진입하였을 때, 즉 하측 픽셀이 흰색일 것을 가정하였다. 이는 통상적으로 영상에서 외곽선을 찾을 때 흰색 픽셀들을 거쳐 흑색 외곽선 픽셀을 만날 때 시작하기 때문이다.

(1) SBF, MSBF, ISBF의 성능 분석

외곽선 알고리즘을 종료하기 위한 판단 기준으로는 두 가지 정보, 즉 추적자의 위치와 방향 정보를 이용하는 것이 일반적이다. 각 알고리즘의 종료조건으로 이 두 가지 정보 중 한 가지, 혹은 두 가지 모두를 사용한다. SBF, MSBF, ISBF는 추적 시작시 시작 정보, 시작 위치와 시작 방향을 기록해두고, 추적을 진행한다. 추적자가 시작 위치에 도달했을 경우, 추적자의 방향 정보가 시작 방향정보와 같으면 종료한다[8]. 이를 동일위치-방향 조건이라고 하자. SBF는 관계없으나, 좌하방향이 인너-아우터 코너인 경우, MSBF와 ISBF에게는 시작조건에 따른 문제점이 존재한다. 즉 좌측과 하측이 흰색이고 좌하픽셀만 흑색이라면 좌하의 인너-아우터 코너와 이에 연결된 외곽선들을 추적한 후 시작점으로 복귀하게 되므로 모든 외곽선을 추적하지 않고 중단된다.

(2) MNT, RSA의 성능 분석

MNT는 기본적으로는 동일위치-방향 조건을 따르지만, 이 조건이 항상 성립하는 것은 아니다. 그림 10은 모두 동일한 위치에서 시작하지만, 초기 방향에 따라 (a)와 같이 시작픽셀 S에서 동일위치-방향 조건에 의해 추적이 종료될 수 있다. 그러나 (b)의 경우에는 S로 돌아왔을 때, 추적자는 시작시와 다른 방향성을 가지는 반면, S 다음 외곽선 픽셀인 A로 전진할 때는 같은 방향성을 가지므로 무한루프를 돌게 될 가능성이 있다. 따라서, 추적자가 추적 후 S로 다시 돌아온 횟수가 n번이라면 이런 경우에도 종료가 가능하여야 한다. 이를 진입횟수 조건이라고 하자. MNT는 이 두 가지 조건중 하나를 만족하면 중단하게 된다.



(a)동일위치-방향 조건 (b)시작위치 도달횟수
그림 10 MNT의 시작 방향에 따른 종료조건

RSA는 추적자의 모든 추적 경로, 즉 추적한 위치 정보를 저장하고, 추적자가 이동시마다 저장한 정보를 조회한다. 즉, 추적자는 전의 위치정보 P_{i-1}과 현재의 위치정보 P_i를 이 저장한 추적 경로와 비교하여 같은 순서로 추적했던 경로가 이미 존재한다면, 추적을 종료하게 된다[9]. 따라서 추적자의 종료가 시작점이 아닌 다른 위치에서 일어나게 된다.

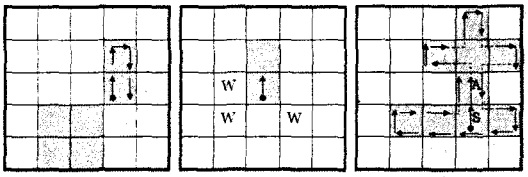
(3) TPA의 성능 분석

TPA는 방향정보가 존재하지만, 동일위치-방향 조건은 만족하지 않으므로 시작위치 도달 횟수조건을 사용하며 종료조건을 삼는다. 원래 알고리즘은 추적자가 시작점에 도착하면 종료하였으나 이 경우 그림 11(a)와 같이 좌측이나 좌하에 픽셀이 존재할 경우 추적하지 못한다. 이는 TPA의 초기 조건인 시작시 좌측 픽셀은 흰색이어야 하는 이유를 보여준다.

이를 개선하기 위하여 Ghuneim[9]은 두 가지 기법을 제안하였는데, 한 가지는 그림 11(b)와 같이 추적자의 좌측, 좌하, 그리고 우하가 반드시 흰색인 경우에만 시작하는 것으로 시작조건을 제한하는 기법이다. 시작조건을 제한하는 경우는 외곽선에서 이러한 픽셀을 찾아야 하므로 실제로 적용하는데 어려움이 많다.

또 다른 기법은 종료 조건으로 3회나 4회이상 진입하였을 때, 추적을 종료하는 기법이다. 그림 11(c)의 경우, 추적자는 S로 돌아온 이후에도 계속 추적을 진행하므로 시작점의 좌측에 있는 영역도 탐색을 계속한다. 그러나 A에 두 번째 진입했을 때 S로 진행하지 않고, B로 바로 이동하므로 종료하기 위해서는 여러 번 외곽선을 추적해야 한다. 따라서 추적의 성능은 좋아지지만, 추적 시간이 늘어나는 문제가 있다.

표 2는 시작조건과 종료조건을 나타낸 것이다. 종료조건으로 가장 바람직한 것은 시작점에서의 동일위치-방향 조건이다. 또한 RSA에서 사용하는 추적 경로 일치 조건의 경우 사실 모든 알고리즘에서 적용 가능하지만, 매



(a) n=1일 경우 (b) 추적 시작 조건 (c) n=2이상일 경우
그림 11 TPA의 시작조건과 종료 조건 (n:시작위치 도달 횟수)

표 2 시작 조건 종료조건 비교
가정: 하측 픽셀은 흰색, 0: 필수, 가능: 적용 가능

	시작점에서 동일위치-방향	시작점 진입횟수	추적 경로 일치	시작조건
SBF	0	가능	가능	없음
MSBF	0	가능	가능	좌는 흰색이고 좌하는 흑색인 경우 회피
ISBF	0	가능	가능	좌는 흰색이고 좌하는 흑색인 경우 회피
MNT	0	0	가능	없음
RSA	X	가능	0	없음
TPA	가능	0	가능	좌는 흰색이어야 함
	동일위치만 적용	가능	가능	좌, 좌하는 흰색이어야 함

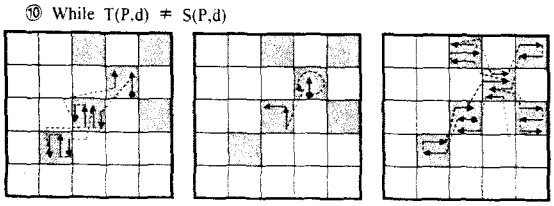
추적 위치마다 추적 경로를 찾는 것은 비효율적이므로 적용을 하지 않는 것이 보다 바람직하다. 시작점 진입횟수 조건 또한 모든 알고리즘에서 가능하나, 이를 너무 낮게 설정하면 추적이 불완전해지고, 3이나 4로 설정하면 추적한 경로를 여러 번 다시 추적하는 경우가 발생한다. 시작조건은 동일위치 조건만 적용할 경우, TPA가 가장 까다롭다. 그리고 MSBF와 ISBF는 좌하의 인너-아우터 코너를 회피해서 시작해야 한다.

4. 외곽선 알고리즘의 개선 방안

4.1 ISBF의 시작 조건 개선 방안

ISBF는 외곽선 추적에 있어서 인너코너를 포함하여 모든 경우를 추적할 수 있으므로 시작조건만 개선하면 된다. 즉, 시작 위치의 좌하에 인너-아우터 코너가 존재하지 않으면 되는 것이다. 가정에 따라 현재 위치는 흑색이고, 하측 방향에는 흰색 픽셀이 존재하므로 좌측이 흰색이고, 좌하가 흑색인 경우만 아니라면 시작 지점으로 삼을 수 있다. 더 간단하게는, 좌하에 인너-아우터 코너가 존재하더라도 시작 방향을 좌측으로 바꾸어주면 해결된다. 최악의 경우는 4방향 모두 인너-아우터 코너가 존재하는 것이다. 이를 해결하기 위해서는 추가적으로 인너-아우터 코너가 존재하지 않을 때까지 좌하 방향으로 이동한 후 시작하면 된다. 따라서 이의 조건에 따른 시작 지점의 프로시저는 다음과 같이 바꿀 수 있다.

① *count* ← 0
② While ($P_{left\text{-}rear}$=black and P_{left}=white and P_{rear}=white)
③ count ← count + 1
④ S(P,d) ← S(P,d_{left})
⑤ if count > 3 then S(P,d) ← S($P_{left\text{-}rear}$,d) and count ← 0
⑥ END While
⑦ T(P,d) ← S(P,d) and tag←0
⑧ Do
⑨ ...
⑩ While T(P,d) ≠ S(P,d)



(a) 시작 조건 개선전 (b)-(c) 제안된 시작 조건의 예
그림 12 ISBF의 제안된 시작 조건

물론 MSBF도 동일하게 시작 조건을 개선할 수 있다. 그림 12는 개선된 시작조건에 의한 추적 과정을 나타낸다.

4.2 TPA의 인너코너 추적 개선 방안

인너코너는 상,하,좌,우 네 방향으로 위치할 수 있는데 TPA는 어떠한 경우에도 인너코너를 탐색하지 못한다. 이를 개선하기 위하여, 좌상 방향에 흑색 픽셀이 존재하는 경우에 그 경로인 상측 픽셀이 흑색인지를 판별하여 추가로 등록한 후 좌상의 픽셀을 등록한다. 또한 좌상과 상이 흰색이고 우상 방향에 흑색 픽셀이 존재하는 경우에는 그 경로인 우측 픽셀이 흑색인지를 판별하여 등록하고, 우상 픽셀을 등록한다. 이러한 경우에는 좌측에 위치한 인너코너 픽셀을 추적하지 못하는 단점이 여전히 존재한다. 따라서 이를 개선하기 위해서 좌상의 흑색 픽셀 추적과정에 좌측 픽셀을 추적하는 과정을 더 추가하였다. 그림 13은 개선된 추적 경로를 나타낸다.

① T(P,d) ← S(P,d) and count_visit_start = 0;
② Do
③ count_rotate ← 0
④ Do
⑤ if $P_{left\text{-}front}$=black
⑥ if P_{left}=black then T(P,d)←T(P_{left},d_{left})
⑦ if $P_{right\text{-}rear}$=black then
⑧ T(P,d)←T($P_{right\text{-}rear}$,d_{right}) and T(P,d)←T(P_{left},d_{left})
⑨ else T(P,d)←T(P_{right},d)
⑩ else if P_{front}=black then
⑪ T(P,d)←T(P_{front},d) and T(P,d)←T(P_{left},d_{left})
⑫ else T(P,d)←T($P_{left\text{-}front}$,d_{left})
⑬ else if P_{front}=black then T(P,d)←T(P_{front},d)
⑭ else if $P_{right\text{-}front}$=black
⑮ if P_{right}=black then
⑯ T(P,d)←T(P_{right},d_{right}) and T(P,d)←T(P_{left},d_{left})
⑰ else T(P,d)←T($P_{right\text{-}front}$,d)
⑱ else T(P,d)←T(P,d_{right}) and count_rotate←count_rotate +1
⑲ While count_rotate <= 3
⑳ if T(P) = S(P) then count_visit_start←count_visit_start+1
㉑ While count_visit_start < n

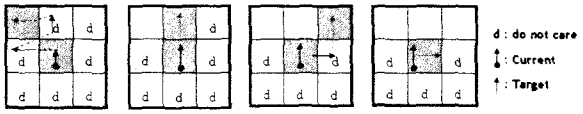


그림 13 인너코너 처리를 위하여 제안한 TPA

5. 실험 결과

각 알고리즘들은 C++ 프로그램을 통해 구현되었으며, 이차원 배열에 의해 구성된 원본이미지로부터 외곽선을 추적하고 그 결과를 제시하였다.

5.1 ISBF의 시작조건 개선 효과

그림 14 (a)는 원본 이미지로서 B는 외곽선 픽셀, 그리고 W는 흰색 픽셀을 나타내며, S는 시작점을 의미한다. 그림들에서 숫자는 해당 픽셀을 지날 때의 흑색 픽셀 경로이고, 아래에 있는 좌표와 방향 정보는 해당 단계의 현재 추적자의 정보이다. ISBF는 현재 위치에서 회전이 일어나는 경우가 있는데, 경로상 숫자는 그 횟수도 포함된 것이며, 같은 위치를 여러번 지날 경우에도 증가한다.

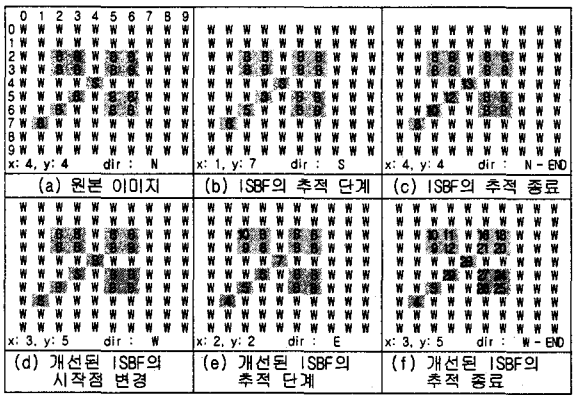


그림 14 제안한 시작 조건 적용 여부에 따른 ISBF 성능 비교

실험을 위해 좌표 (4,4)에 N방향을 가진 추적자가 위치하도록 한 후 기존 방법과 제안한 방법을 비교하였다. 실험 결과 기존의 ISBF는 그림 4(b),(c)와 같이 추적자의 좌하쪽 픽셀들을 추적한 후, 시작점에서 종료되었다. 그러나 제안한 기법을 적용한 ISBF는 시작조건에 따라 시작점을 (3,5)로 이동한 후 W 방향으로 시작하였고 모든 외곽선을 추적하였다. 그러나 모든 이미지 픽셀이 서양 장기판 모양으로 배치된 경우에는 다른 알고리즘들과 마찬가지로 가장 바깥쪽의 픽셀들만 추적할 수 있다.

5.2 제안한 TPA의 인너코너 추적 결과

인접코너는 좌,우,상,하 네 가지 경우가 있는데, 이중 하측에 위치한 경우는 추적자 알고리즘의 공통 가정에 의해 배제하였다. 그림 15는 TPA의 제안한 기법으로 추적한 인너코너 이미지들의 추적결과이다. 그림 15 (a)-(c)는 인접코너를 가진 원본 이미지이고, (d)-(f)는 각각의 추적결과이다. 실험 결과 제안한 알고리즘은 인접코너들을 모두 추적하였으나, 그림 15 (b)와 같은 패턴인 경우, 즉 좌측에 위치한 인너코너일 때, 추적자가 3, 5, 7 픽셀에서 우측 인접픽셀들을 찾지 못하는 경우가 발생하였다. 이는 TPA의 가정인 시작시 좌측의 픽셀은 흰색이어야 한다는 조건을 여전히 따라야 한다는 것을 보여준다.

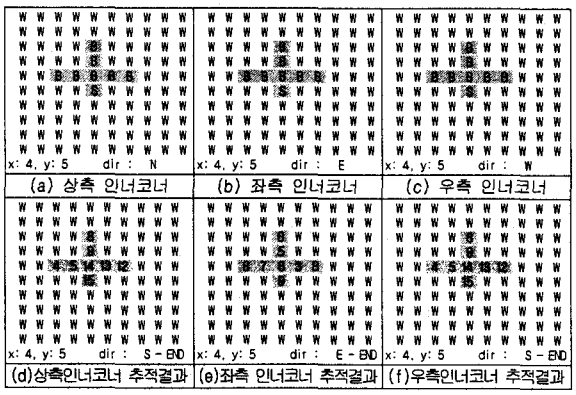


그림 15 제안한 기법을 적용한 TPA의 인너코너 추적 성능

6. 결론

본 논문에서는 기존의 외곽선 추적 알고리즘인 SBF, MSBF, ISBF, MNT, RSA, 그리고 TPA의 알고리즘들의 성능을 연결관계와 시작종료 조건을 중심으로 분석하였다. 분석결과 SBF는 가장 단순한 반면, 인너코너와 인너-아우터 코너 추적이 불가능하였다. MSBF 또한 상측 인너 코너에서 추적이 불가능하였고 시작조건에 제한이 있었다. MNT와 RSA는 알고리즘이 유사하며, 모든 인너코너 추적이 불가능했다. TPA는 시작조건이 까다롭고, 인너코너 감지가 불가능했다. 이중에서 모든 경우의 외곽선을 추적할 수 있는 것은 ISBF이었는데, 다만 MSBF와 마찬가지로 시작시 좌하 인너-아우터 코너가 있는 경우를 회피해야하는 제약 조건이 있었다.

이러한 문제점들을 해결하기 위해서 먼저 ISBF의 시작조건을 개선하는 기법을 제안하였다. 이 기법을 이용하면 시작조건의 제약을 받지 않고, 직선형, 인너코너, 인너-아우터 코너, 그리고 아우터 코너 모두를 추적할 수 있었다. 부가적으로 TPA의 인너코너 추적 불가능 문제 해결을 위해 기존의 알고리즘을 개선할 수 있는 기법을 제안하였다. 이 역시 실험 결과 실제로 모든 경우의 인너코너를 추적할 수 있었다. 그러나 추적자의 좌측에 인너코너가 배치된 경우에는 추적자의 우측 픽셀들을 추적하지 못하는 문제점은 해결하지 못했다. 따라서 시작조건이 개선된 ISBF를 사용하는 것이 바람직하다.

감사의 글

이 논문은 2006년도 한국과학재단 특정기초연구(R01-2005-000-10898-0)의 연구비 지원으로 이루어졌습니다.

참고 문헌

[1] S. Marchand-Maillet, Y. M. Sharaiha, *Binary Digital Image Processing*, pp.173-180, Academic Press, 2000.

[2] W. A. McQueen, *Contour tracing and boundary detection for object*, United States Patent 6,674,904, Jan. 6, 2004.

[3] I. Pitas, *Digital Image Processing Algorithms and Applications*, pp.275-329, John Wiley & Sons, 2000.

[4] E. Gose, R. Johnsonbaugh, S. Jost, *Pattern recognition and Image analysis*, pp.338-346, Prentice Hall, 1996.

[5] F. Chang, C.-J. Chen, and C.-J. Lu, "A linear-time component-labeling algorithm using contour tracing technique," *Comput. Vis. Image Underst.* 93(2), pp. 206-220, 2004.

[6] M. Das, M. J. Paulik, N. K. Loh, "A Bivariate Autoregressive Modeling Technique for Analysis and Classification of Planar Shapes," *IEEE Transaction on Pattern Analysis and Machine Intelligence*, Vol. 12, No. 1, pp. 97-103, Jan. 1990.

[7] S. Papert, *Uses of technology to enhance education*, AI memo 298, AI Lab., MIT, 1973: ftp://publications.ai.mit.edu/ai-publications/pdf/AIM-298.pdf, Aug., 2006.

[8] 정철호, 한탁돈, "개선된 간단한 경계선 추적자 알고리즘," *정보과학회논문지:소프트웨어 및 응용*, 33권 4호, pp.427-439, 2006년 4월.

[9] A. G. Ghuneim, "Contour Tracing," http://www.imageprocessingplace.com/DIP/dip_downloads/tutorials/contour_tracing_Abeer_George_Ghuneim/index.html, Aug., 2006.

[10] A. Mirante, N. Weingarten. "The radial sweep algorithm for constructing Triangular Irregular Networks," *IEEE Computer Graphics and Applications*, pp.11-21, 1982.

[11] T. Pavlidis, *Algorithms for Graphics and Image Processing, Computer Science Press*, Rockville, Maryland, 1982.