# 단계적 후보 축소에 의한 예제기반 초해상도 영상복원을 위한 고속 패치 검색

## (Fast Patch Retrieval for Example-based Super Resolution by Multi-phase Candidate Reduction)

박 규 로 † 김 인 중 ††
(Gyuro Park) (In-Jung Kim)

**요 약** 예제기반 초해상도 영상복원은 영상 패치의 대한 학습 및 검색을 통해 저해상도 영상으로부터 고해상도 영상을 복원하는 방법으로써 성능이 좋고 한 장의 저해상도 영상에 대하여도 적용 가능하다. 그러나 복원 과정에서 패치 검색에 많은 비교 연산이 요구되기 때문에 속도가 매우 느리다. 복원 속도를 향상시키기 위해서는 효과적인 패치 검색 알고리즘이 요구된다. 본 논문에서는 패치 검색에 사용 가능한 다양한 고차원 특징 검색 방법들을 실제 초해상도 영상복원 시스템에 적용하여 그 성능을 비교하였다. 또한 문자 인식 분야에서 성공적으로 적용되어왔으나 초해상도 영상복원에서는 사용되지 않았던 단계적 후보 축소 방법을 패치 검색 단계에 적용할 것을 제안한다. 실험 결과 기존의 방법 중에서는 LSH가 가장 좋은 성능을 나타내었다. 본 논문에서 제안한 단계적 후보 축소에 의한 패치 검색 방법은 LSH보다 더욱 우수하여 1024×1024 영상의 복원 시 LSH보다 최대 3.12배 빠른 복원 속도를 나타내었다.

**키워드** : 예제기반 초해상도 영상 복원, 고속 패치 검색, 단계적 후보축소, LSH

***Abstract*** Example-based super resolution is a method to restore a high resolution image from low resolution images through training and retrieval of image patches. It is not only good in its performance but also available for a single frame low-resolution image. However, its time complexity is very high because it requires lots of comparisons to retrieve image patches in restoration process. In order to improve the restoration speed, an efficient patch retrieval algorithm is essential. In this paper, we applied various high-dimensional feature retrieval methods, available for the patch retrieval, to a practical example-based super resolution system and compared their speed. As well, we propose to apply the multi-phase candidate reduction approach to the patch retrieval process, which was successfully applied in character recognition fields but not used for the super resolution. In the experiments, LSH was the fastest among conventional methods. The multi-phase candidate reduction method, proposed in this paper, was even faster than LSH: For 1024×1024 images, it was 3.12 times faster than LSH.

**Key words** : Example-based super resolution, Fast Patch Retrieval, Multi-phase Candidate Reduction, LSH

· 이 논문은 2007년 정보(교육과학기술부)의 재원으로 한국연구재단(구 학술진흥재단)의지원을 받아 수행된 연구임(KRF-2007-331-D00420)
· 이 연구는 2009년도 한동대학교 교내연구지원사업에 의한 것임

† 학생회원 : 한동대학교 정보통신공학과
outzzang@gmail.com
†† 종신회원 : 한동대학교 전산전자공학부 교수
ijkim@handong.edu

논문접수 : 2009년 1월 14일
심사완료 : 2010년 1월 26일


정보과학회논문지 : 소프트웨어 및 응용 제37권 제4호(2010.4)

## 1. 서 론

오늘날 컴퓨터 비전 및 패턴인식 기술은 기술적으로 많은 발전을 이루었으며, 다양한 분야에 성공적으로 응용되고 있다. 그러나 통제되지 않은 환경에서 획득한 영상은 화질 및 해상도의 저하로 인해 인식 기술을 적용하기 어려운 경우가 많다. 그 동안 영상처리 기술을 이용하여 이러한 한계를 극복하기 위한 노력이 진행되어 왔다.

초해상도 영상복원(super-resolution image reconstruction)은 동일 장면을 촬영한 한 장 또는 여러 장의

저해상도 영상으로부터 세부 형태를 복원하여 고해상도 영상을 생성하는 기술로써, 그 효과가 우수하다고 알려져 있다[1-3]. 특히, 예제기반 초해상도 영상복원은 성능이 우수할 뿐 아니라, 한 장의 저해상도 영상만으로도 고해상도 영상을 복원할 수 있다[4-7]. 그러나, 복원 속도가 매우 느리고, 입력 영상의 특성이 학습 영상과 이질적인 경우 복원 성능이 크게 저하되는 단점이 있다[7]. 특히, 느린 복원 속도는 예제기반 초해상도 영상복원을 다양한 분야에 실용화하기 위해서는 반드시 극복되어야 한다.

예제기반 초해상도 영상복원은 영상을 패치 단위로 분리한 후 패치에 대한 학습 및 검색을 이용하여 영상을 복원해내는 기술이다. 예제기반 초해상도 영상복원에서 복원 시간의 대부분은 패치 검색 과정에서 소요된다. 따라서, 복원 속도의 개선을 위해서는 효과적인 패치 검색 방법이 필수적이다. 영상 패치는 고차원 특징으로 볼 수 있기 때문에 패치 검색 문제는 고차원 특징 검색의 한 종류로 생각할 수 있다. 따라서, 고차원 특징 검색 분야에서 제안된 효율적인 방법들을 예제기반 초해상도 영상복원에 적용할 경우 검색 속도를 개선할 수 있다.

본 연구에서는 고차원 특징 검색 방법들 중 성능이 우수하다고 알려진 방법들을 실제 예제기반 초해상도 영상복원 시스템에 적용하여 각각의 복원 속도를 실험을 통해 비교하였다. 뿐만 아니라, 문자인식 분야에 성공적으로 적용되어 온 단계적 후보 축소 방법을 예제기반 초해상도 영상복원의 패치 검색에 적용하여 상당한 속도 개선 효과를 얻었다. 실험결과 패치 검색 방법은 영상 복원 속도에 큰 영향을 미쳤으며, 기존 방법들 중에는 LSH의 성능이 가장 우수하였다. 본 논문에서 제안한 단계적 후보 축소 방법은 LSH 보다 더욱 우수한 성능을 나타내었는데, $1024 \times 1024$ 영상의 복원 시 LSH 보다 최대 3.12배 빠른 복원 속도를 나타내었다.

본 논문의 2장에서는 예제기반 초해상도 복원 방법과 고차원 특징 검색을 위한 기존의 방법들을 소개한다. 3장에서는 본 연구에서 제안한 단계적 후보 축소에 의한 패치 검색 방법을 설명한다. 4장에서는 제안하는 방법과 다양한 기존 검색 방법들을 실제 예제기반 초해상도 복원 시스템에 적용하여 비교한 실험 결과를 제시한다. 마지막으로 5장에서는 결론을 맺는다.

## 2. 기존 연구

### 2.1 예제기반 초해상도 영상복원

영상은 다양한 주파수의 성분으로 이루어지는데, 해상도 저하 시 주로 소실되는 성분은 주로 세부 형태를 반영하는 고주파 성분이다[4]. 저해상도 영상을 단순 보간에 의해 확대한다면 영상의 크기는 증가하나 고주파 성분은 복원되지 않는다. 그러나, 다수의 고해상도 학습 영상으로부터 고주파 성분을 미리 학습하여 복원 과정에 사용한다면 소실된 고주파 성분을 복원함으로써 영상의 크기뿐 아니라 세부 형태까지도 복원할 수 있다.

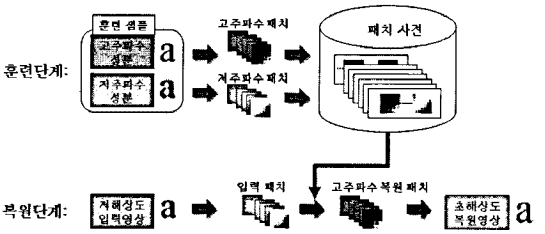


그림 1 예제기반 초해상도 영상복원 시스템

예제기반 초해상도 영상복원 방법은 그림 1과 같이 패치 학습 단계와 영상 복원 단계로 구성된다. 학습 단계에서는 고해상도 학습 영상을 고주파 성분과 저주파 성분으로 분리한 후 각 성분을 패치로 분할한다. 그리고, 각 고주파수 패치와 그에 대응하는 저주파수 패치를 패치 쌍의 형태로 패치사전에 저장한다. 영상 복원 단계에서는 패치사전을 이용하여 저해상도 입력 영상에 소실된 고주파수 성분을 복원함으로써 고해상도 영상을 생성한다[4,7].

본 연구에서는 Freeman의 방법을 기반으로 시스템을 구성하기 때문에 Freeman의 방법을 중심으로 소개하겠다[4]. 그러나, 다른 예제기반 초해상도 영상복원 방법들도 패치의 학습 및 검색을 이용해 구현되기 때문에, 본 연구 결과는 Freeman의 방법에만 국한되지 않는다[4,5,7].

패치 학습 단계에서는 먼저 고해상도 학습 영상을 FFT에 의해 저주파 성분과 고주파 성분으로 분리한다. 그 후 양쪽 성분의 동일 위치에서 추출한 패치를 패치 쌍 ($H_s$, $L_s$)으로 구성하여 패치사전 T에 저장한다.

$$T = \{T_s = (H_s, L_s) \mid H_s : \text{고주파수 패치}, \; L_s : \text{저주파수 패치}, s = 1,...,n\}$$

각 패치 쌍에서 $H_S$는 영상 복원 시 사용될 고주파수 성분을 제공하는 역할을 하며, $L_S$는 각 입력 패치의 복원에 사용할 고주파수 패치 $H_S$를 검색하기 위한 검색 키로 사용된다. 그런데, 패치 키를 저주파수 성분만으로 구성할 경우 적합한 검색 결과를 얻기 어렵다. [4]에서는 이러한 문제점을 극복하기 위하여 저주파 성분 외에도 고주파 성분의 일부를 검색 키에 포함시켰다.

영상 복원 단계에서는 먼저 보간법을 이용하여 저해상도 입력 영상을 목표하는 크기로 확대한다. 그리고, 확대 영상을 FFT에 의해 고주파 성분과 저주파 성분으로 분리한 후, 저주파 성분을 분할하여 입력 패치들을 추출한다. 그리고, 각 입력 패치 $I_i$와 최소 거리를 갖는 $L_i$를 패치사전에서 검색한 후, $L_i$와 쌍을 이루는 고주

파 패치 $H_i$를 입력 패치 $I_i$와 결합하여 복원 패치를 생성한다. 마지막으로, 복원 패치들을 연결하여 고해상도 영상을 생성한다.

### 2.2 고차원 특징 검색

고차원 특징 검색이란 고차원 특징 공간 상의 객체들로 이루어진 검색 공간에서 입력된 특징 벡터와 가장 가까운 거리에 있는 객체를 찾는 문제이다. 그 동안 고차원 특징 검색은 지리정보 시스템(GIS), 객체 인식, 다중 데이터베이스 등 다양한 문제에 적용되어 왔다[8]. 고차원 특징 검색은 크게 최적 검색(optimal search)과 유사 최적 검색(suboptimal search)으로 나눌 수 있는데, 각 종류에 속한 방법들은 다음과 같다.

2.2.1 최적 검색(optimal search)

최적 검색은 검색 공간의 객체 중 주어진 검색 대상에 가장 가까운 객체를 찾는 방법이다. 최적의 검색 결과를 보장하지만, 객체의 수가 많고 특징의 차원이 높을 경우 속도가 느리다. 대표적인 최적 검색 방법에는 선형 검색(linear search), 해싱(hashing), kd-트리 검색 등이 있다[8,9].

• 선형 검색

선형 검색은 모든 패치 쌍에 대해서 순차적 검색을 통해 최적 검색 결과를 찾는 방법이다. 선형 검색에서는 패치사전 내의 모든 패치 쌍을 입력 패치와 직접 비교하기 때문에 대용량 패치사전에 적용할 경우 속도가 매우 느리다. 그러나, 각 패치에 대하여 최적의 검색 결과를 제공하기 때문에, 본 연구에서는 타 방법들로 복원한 영상의 화질을 평가하기 위한 기준으로 사용하였다.

• 해싱(Hashing)

해싱은 해시 함수를 통해 검색 대상의 위치를 직접 계산하기 때문에 속도가 빠르며, 저차원 특징에 대해서는 매우 효과적이다. 단, 객체의 수가 많고 특징의 차원이 높을 경우 성능이 저하되는 문제점이 있다[10,11].

• kd-트리 검색[9]

kd-트리는 $R^k$ 공간상의 객체들로 이루어진 이진 트리의 일종이다. kd-트리에서 L번째 레벨은 L % k번째 차원에 대응된다. 각 레벨에서는 그와 대응하는 차원의 중간 값을 기준으로 객체들을 양분하여 각각 좌우의 서브 트리를 재귀적으로 구성함으로써 이진 트리를 구성한다. kd-트리 생성 알고리즘은 표 1과 같다.

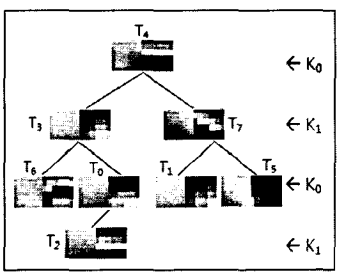
표 1의 알고리즘을 표 2의 2차원 패치사전에 적용하여 생성한 kd-트리는 그림 2(a)와 같다. 실제 패치는 고차원 벡터이나 표 2와 그림 2에서는 간단히 설명하기 위해 2차원 벡터로 표시하였다. (b)는 kd-트리의 생성 과정에서 각 노드에 의해 분할된 좌표계이다. 각 수직선과 수평선은 각 레벨에서 객체들을 양분하는데 사용된 중간 값의 좌표를 의미하는데, 이는 (a)의 각 중간 노드

표 1 kd-트리 생성 알고리즘

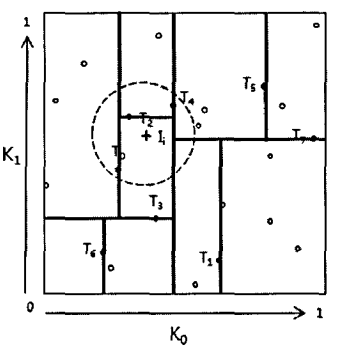
```
Initialization
  n = number of points
  k = data dimension
  points = data array [0,...,n-1]

Algorithm kdtree (points, depth)
{
  if (points is empty)
    return null
  else
    axis = depth % k
    SortByAxis (points, axis)
    median = (number of points) / 2
    kdtree_node node
    node.location = median
    node.leftChild = kdtree (points[0:median], depth+1)
    node.rightChild = kdtree (points[median+1:n-1], depth+1)
    return node
  end if
}
```

표 2 2차원 패치사전의 예

| | $T_0$ | $T_1$ | $T_2$ | $T_3$ | $T_4$ | $T_5$ | $T_6$ | $T_7$ |
|---|---|---|---|---|---|---|---|---|
| K0 | 0.25 | 0.65 | 0.3 | 0.4 | 0.45 | 0.8 | 0.2 | 0.9 |
| K1 | 0.45 | 0.15 | 0.65 | 0.3 | 0.7 | 0.75 | 0.2 | 0.55 |




(a) kd-트리




(b) kd-트리에 의해 분할된 좌표계

그림 2 표 2의 데이터로 구성한 kd-트리

에 대응된다. (b)에서 각 노드($T_0, \cdots, T_7$)에 의해 분할된 사각형 영역(hyper-rectangle)를 빈(bin)이라고 하는데, 이는 kd-트리에서 각 서브트리에 해당한다.

특정 검색 대상이 주어졌을 때 kd-트리 내에서 최소 거리를 갖는 객체에 대한 검색은 다음과 같이 이루어진다. 먼저 이진 검색 트리(binary search tree)와 동일한 방법으로 루트 노드부터 잎 노드까지 검색을 수행한다. 그리고, 검색 과정에서 방문한 노드 중 검색 대상과 가장 가까운 노드를 최적 노드의 초기값으로 저장한다. 그 후, 백트래킹을 통해서 방문하지 않은 빈들을 검색한다. 이 때, 검색 대상으로부터의 최소 거리가 기존의 최적 노드보다 먼 빈들은 검색에서 제외함으로써 검색 공간을 축소한다. 각 빈을 검색하는 도중 최적 노드보다 검색 대상에 더 가까운 노드가 발견될 경우 그 노드를 새로운 최적 노드로 선택한다. 현재의 최적 노드보다 가까운 거리에 있는 빈이 더 이상 없을 경우 검색을 종료한다.

kd-트리는 저차원 특징에 대해서는 O(log n)에 근접한 우수한 성능을 나타내지만, 트리가 불균형하거나 고차원 특징일 경우 O(n)에 가까운 낮은 성능을 보인다. 또한, 특정 빈에 객체가 편중되는 문제도 발생한다[12].

2.2.2 유사 최적 검색(suboptimal search)

유사 최적 검색은 검색 결과의 정확도를 다소 희생함으로써 검색 속도를 개선한 검색 방법이다. 검색 대상 x와 가장 근접한 객체와의 거리를 r이라고 할 때 유사 최적 검색 방법은 x와의 거리가 최대 r의 c배 ($c = 1 + \varepsilon$, $\varepsilon > 0$) 이내인 객체를 빠른 시간 내에 찾는 방법이다. 유사 최적 검색 방법에는 kd-트리를 이용한 유사 검색, BBF(Best Bin First) 검색, LSH(Locality Sensitive Hashing) 등이 있다[9,12,13]. 유사 최적 검색 방법을 사용할 경우 속도와 정확도 간의 타협을 통해 사용 환경에 적합한 성능을 얻어낼 수 있다.

• BBF 검색[12]

BBF검색은 kd-트리의 검색 알고리즘으로부터 고차원 특징에 대하여 검색 효율이 저하되는 단점을 보완한 방법이다. 고차원 검색 시 kd-트리의 성능이 저하되는 주 원인은 검색 대상에서의 거리가 그 순간의 최적 노드보다 매우 근소하게 가까운 빈들이 많아 탐색 공간의 많이 축소되지 않는 것이다. 이러한 문제를 극복하기 위해 BBF검색에서는 두 가지 면에서 수정을 가하였다.

먼저, 백트래킹 시 방문하는 노드의 총 개수를 제한함으로써 검색 시간이 과도하게 소모되는 것을 방지하였다. 그 결과, 최적의 검색 결과는 보장하지는 못하게 되었으나, 검색 속도는 크게 향상되었다. 둘째, 백트래킹 단계에서 kd-트리 검색 알고리즘이 트리 구조의 연결 관계를 따라 노드를 검색하는 반면, BBF검색 알고리즘은 우선순위 큐를 사용하여 검색 대상과의 거리 순으로 검색한다. 하향 검색 과정에서 각 노드로부터 특정 자식 노드로 이동할 때 반대쪽 자식 노드를 우선순위 큐에 저장한다. 이 때, 우선순위 큐 내에서 노드의 키는 그 검색 대상으로부터의 거리로 지정한다. 백트래킹 시 키 값이 작은 노드부터 방문함으로써 검색 대상과 가까운 빈들을 먼저 검색한다.

• LSH[13]

LSH는 특징 벡터간 유사성이 보존되도록 설계된 해시 함수를 통해 최소거리의 객체가 검색 대상 특징 벡터와 동일한 해시 함수 값을 가질 확률이 높도록 개선한 해싱 방법이다.

먼저 d차원의 특징 벡터 $p = (x_1, \cdots, x_d)$을 다음과 같은 변환 함수 v(p)을 이용하여 d'차원의 해밍 공간으로 변환한다.

$$p' = v(p) = Unary_C(x_1)..Unary_C(x_d)$$

여기에서 d'는 C*d로 계산된다. C는 $x_i$가 가질 수 있는 최대값이다. $Unary_C(x)$를 x개의 1과 (C-x)개의 0으로 이루어진 길이 C인 비트열을 나타낸다.

이와 같이 v(p)를 정의할 경우 두 특징 벡터 p, q의 변환 벡터 p'와 q'간의 해밍 거리는 p와 q간의 $L_1$-norm와 동일하게 되어 다음과 같은 관계가 성립한다. $D_1$은 $L_1$-norm 거리, $D_h$는 해밍 거리를 나타낸다.

$$D_1(p', q') = D_h(v(p'), v(q'))$$

특징 벡터 p의 함수 $h_i(p)$를 변환 벡터 p'의 i번째 비트라고 할 때, 두 특징 p, q에 대하여 $h_i(p)$, $h_i(q)$가 같을 확률 $\Pr[h_i(p)=h_i(q)]$는 다음과 같이 계산된다.

$$\Pr[h_i(p) = h_i(q)] = 1 - D_h(p,q)/d'$$

이 때 주어진 네 상수 (r, cr, $P_1$, $P_2$)에 대해서 다음 조건을 만족하는 $h_i$의 집합을 H라고 하자.

$$If \ \|p-q\| \le r, then \ \Pr[h_i(p) = h_i(q)] \ge P_1$$

$$If \ \|p-q\| \ge cr, then \ \Pr[h_i(p) = h_i(q)] \le P_2$$

LSH에서는 복수의 해시 테이블을 사용하는데, $\ell$개의 테이블 중 j번째 테이블 $\tau_j \ (1 \le j \le l)$의 함수 $g_j$는 다음과 같이 정의한다.

$$g_j(p) = [h_1(p), h_2(p), ... h_b(p)]^T, \ h \in H, 1 \le j \le \ell$$

이 때, 각각의 해시 테이블 $\tau_j$를 LSHT(Locality-Sensitive Hash Table)이라고 한다. b는 각 해시 함수가 갖는 비트 수이다. b와 $\ell$는 r과 cr의 값에 따라서 결정되는데, 그 방법은 [13]을 참조하기 바란다. 검색 과정에서는 검색 대상 q가 주어졌을 때 $\ell$개의 LSHT에 대하여 각각 $g_j(q)$번째 버켓들을 선형 검색함으로써 검색 결과를 출력한다.

## 3. 단계적 후보 축소에 의한 검색

단계적 후보 축소(Multi-phase Candidate Reduction) 방법은 패치 검색을 포함한 고차원 특징 검색 방법으로는 잘 사용되지 않지만, 한자와 같이 글자 수가 많은 언어의 인식에 성공적으로 적용된 바 있다[14]. 본 연구에서는 이 방법을 예제기반 초해상도 영상복원의 패치 검색 단계에 적용할 것을 제안한다. 단계적 후보 축소에 의한 검색 방법은 그림 3과 같이 군집화(clustering)에 의한 후보 선택, 개략 분류에 의한 후보 축소, 그리고 선형 검색에 의한 상세 분류 등 세 단계로 구성된다.

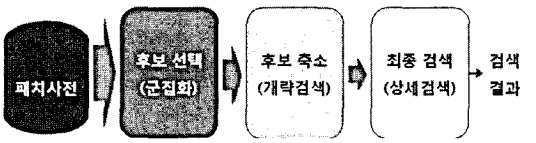



그림 3 단계적 후보 축소에 의한 검색

### 3.1 군집화에 의한 후보 선택

후보 선택 단계는 방대한 크기의 패치사전 중에서 검색 대상에 비교적 가까운 객체, 즉 패치 쌍들을 검색 후보로 선택하고 그 외의 패치 쌍들은 검색에서 제외함으로써 검색 공간을 축소하는 단계이다. 후보 선택 단계에서는 속도, 정확도, 그리고 효율성이 중요하다. 후보 선택 방법의 속도는 계산량에 반비례한다. 정확도는 선택된 후보 내에 최소 거리 객체가 포함될 확률이다. 효율성은 정확도를 유지하면서도 적은 수의 후보를 선택할수록 우수하다. 문자 인식 분야에서는 군집화에 의한 후보 선택 방법이 우수한 성능을 나타내었다[14].

학습 단계에서는 패치 쌍들을 서로 가까운 것들끼리 군집화한다. 대표적인 군집화 알고리즘으로는 표 3과 같은 k-means 알고리즘이 있다. 표 3에서 $o_j$는 각 단계에서 학습에 사용되는 객체를 나타내며 $m_i$는 i번째 군집의 대표값을 의미한다. distance($o_j$,$m_i$)는 $o_j$와 $m_i$간의 거리를 나타낸다. k-means 알고리즘은 각 객체를 가장 가까운 군집에 추가하고 새롭게 구성된 군집의 대표값 $m_i$을 갱신하는 과정을 반복함으로써 유사한 객체끼리 군집화한다.

그러나, k-means 알고리즘을 고차원 특징 벡터에 적용할 경우 객체들이 특정 군집으로 편중되는 문제가 발생한다. Ahalt는 이러한 문제를 극복하기 위하여 각 객체가 속할 군집을 찾는 과정에 각 군집의 크기에 비례하는 벌점을 가하는 FSCL(Frequency Sensitive Competitive Learning) 알고리즘을 제안하였다[15]. FSCL에서는 표 3에서 밑줄로 표시한 6번째 행을 다음과 같이 수정한다.

표 3 k-means 알고리즘

```
Algorithm k-means (objects, k)
{
    initialize clusters {Ci | 1 ≤ i ≤ k} with arbitrary k objects.
    for all mi's.
      mi = mean of objects in Ci.
    repeat until all clusters converge
      for each object oj
        i* = argmini distance(oj, mi)
        insert oj into Ci*
      end for
      for all mi's
        mi = mean of objects in Ci
    end repeat
}
```

$$i^* = \text{argmin}_i \ \text{distance}(o_j, m_i)\text{Penalty}(C_i)$$

여기에서 $Penalty(C_i)$는 $C_i$의 크기에 따른 벌점을 나타낸다. 본 연구에서도 FSCL을 적용하였으며 $Penalty(C_i)$로는 다음과 같이 군집 크기의 제곱근을 사용하였다. $|C_i|$는 군집 $C_i$의 크기를 의미한다.

$$\text{Penalty}(C_i) = \sqrt{|C_i|}$$

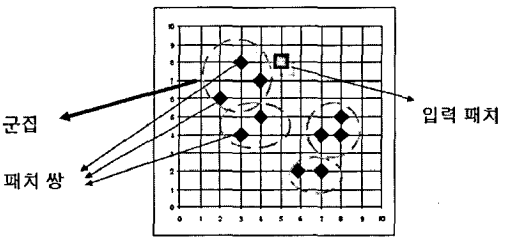



그림 4 군집화에 의한 후보 선택

검색 대상이 입력되면 각 군집의 대표값 $m_i$과 검색 대상의 거리를 측정하여 그 중 가장 가까운 u개의 군집을 선택한다. 그리고 선택된 군집에 속하는 객체들을 검색 후보로 선택한다. u는 실험적으로 가장 우수한 값을 사용하였으며 군집의 수 k는 [14]에서 제안한 바와 같이 $\sqrt{un}$으로 결정하였다. n은 전체 객체의 수이다.

### 3.2 개략 검색에 의한 후보 축소

앞 절에서 설명한 후보 선택 단계는 많은 패치를 검색에서 제외시킴으로써 속도를 향상시킨다. 그러나, 대용량의 패치사전에 대하여 검색의 정확도를 유지하기 위해 큰 u를 사용할 경우 적지 않은 수의 후보를 생성한다. 따라서, 후보 축소를 통한 추가적인 속도 개선이 요구된다.

패치 검색에 많은 시간이 소모되는 또 다른 이유는 패

치가 고차원 특징 벡터이기 때문이다. 비교 과정에서 특징 차원을 축소한다면 비교에 필요한 연산을 줄일 수 있다. 차원을 축소하면 정보의 손실이 발생하지만, 최종 검색 단계가 아닌 후보 축소 단계에서는 복수의 후보를 선택하기 때문에 심각한 성능 저하를 발생하지는 않는다.

차원 축소 과정에 PCA(Principal Component Analysis)를 이용하면 정보의 손실을 최소화 할 수 있다. PCA를 이용해 D차원 벡터 X를 d차원만을 이용해 $\tilde{X}$으로 근사하는 방법은 다음과 같다.

$$\tilde{X} = \mu + \sum_{i=1}^{d} a_i e_i$$

여기에서 μ는 X의 평균이다. $e_i$ ($1 \le i \le d$)는 X의 공분산 행렬의 고유 벡터(eigen vector) 중 고유값(eigen value)이 큰 d개의 벡터이다. $a_i$는 입력 벡터 X의 주성분(principal component)인데, (X-μ)와 $e_i$의 내적에 의해 얻을 수 있다. d차원으로 축소된 변환 벡터 Z는 다음과 같이 구성된다.

$$Z = (a_1, a_2, ..., a_d)^T$$

PCA를 이용한 차원 축소는 패치 검색에 다음과 같이 적용된다. 패치 학습 단계에서 패치 쌍의 검색 키에 대하여 공분산 행렬의 고유 벡터를 계산한 후, 각 패치 쌍의 검색 키를 변환 벡터의 형태로 저장한다. 검색 대상이 입력되면 PCA에 의해 변환 벡터를 생성한다. 그리고, 후보 선택 단계에서 선택된 $n_1$개의 검색 후보와 각각 비교하여 가장 가까운 $n_2$개의 패치 쌍만을 최종 상세 검색 단계에 전달한다.

### 3.3 최종 상세 검색

상세 검색 단계는 검색 후보 생성 및 축소 과정을 통해 결정된 $n_2$개의 후보들을 각각 검색 대상과 비교하여 최종적인 검색 결과를 결정하는 단계이다. 하나의 검색 결과만을 선택하기 때문에 높은 정확도를 얻기 위해서는 정보의 손실을 최소화해야 한다. 따라서, 최적 검색 방법인 선형검색을 사용하였고, 비교 연산에 고차원 특징 벡터를 사용하여 정보의 손실을 방지하였다. 단, PCA를 통해 소수의 차원만을 축소할 경우 정보의 손실이 미미하기 때문에 정확도 저하를 무시할 만한 수준으로 제한하면서도 약간의 속도 개선 효과를 얻을 수 있다. 따라서 본 연구에서는 $d_2$ 차원의 변환 벡터를 이용하여 최종 상세 검색을 수행하였다. ($d < d_2 < D$)

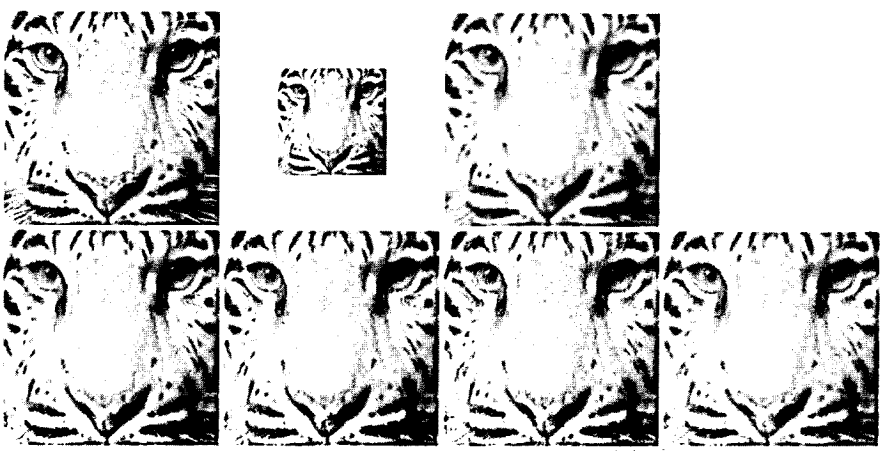
## 4. 실 험

### 4.1 실험 내용

본 실험에서는 2.2절에서 소개한 다양한 검색 방법과 3장에서 제안한 단계적 후보 축소에 의한 검색 방법을 2.1절에서 소개한 초해상도 영상복원 시스템에 적용하여 실제 영상 복원 속도를 비교 평가하였다. 실험은 Q6600 2.4GHz CPU가 설치된 PC에서 실시하였다.

본 실험에서 비교한 방법 중 유사 최적 검색 방법들은 파라미터에 의해 속도와 복원 영상의 오차율을 제어할 수 있다. 이 때, 복원 속도는 허용 오차율에 반비례하는데, 본 실험에서는 검색 방법의 최대 허용 오차율을 1%, 5%, 10% 등 정해진 값으로 제한시키면서 화질 변화에 따른 복원 속도의 변화를 측정하였다. 그림 5는 축소 영상 (b)에 대해서 예제기반 초해상도 영상복원 시스템을 이용하여 최대 허용 오차율 별 복원 영상들의 예이다.

실험 데이터로는 인터넷에서 수집한 자연 영상 8장과 카메라로 촬영한 텍스트 영상 2장 등 모두 10장을 사용하였다. 각 영상에 대하여 크기 별 복원 속도를 측정하기 위하여 64 × 64, 128 × 128, 256 × 256, 512 × 512, 1024 × 1024 등 5단계로 크기를 조절하여 총 50장의 영



| (a) | (b) | (c) | |
|---|---|---|---|
| (d) | (e) | (f) | (g) |

(a) 원본 영상
(b) 축소 영상
(c) Bicubic 보간 영상
(d) 선형검색 복원 영상
(e) 오차율 1% 복원 영상
(f) 오차율 5% 복원 영상
(g) 오차율 10% 복원 영상

그림 5 실험 영상 예

상을 사용하였다. 실험 방법은 50장의 실험 데이터를 먼저 가로와 세로를 절반크기로 축소한 후 다시 원래의 크기로 복원을 수행하며 시간을 측정하였다.

복원 영상의 화질 비교에 있어서 패치 검색 외에 다른 요인들의 영향력을 배제하기 위하여 최적 검색 방법인 선형 검색에 의한 복원 영상을 기준으로 화질을 평가하였다. 선형 검색을 이용한 복원 영상과 각 패치 검색 별 복원 영상간의 비교 척도로는 MSSIM(Mean Structural Similarity Index Metric)를 사용했는데, 이는 널리 사용되는 PSNR(Peak Signal-to-Noise Ratio)이나 MSE(Mean Squared Error)보다 영상의 직관적인 차이를 더욱 잘 반영하는 것으로 알려져 있다[16].

### 4.2 실험 파라미터

본 실험에 사용한 파라미터는 사전 실험을 통해 좋은 결과를 나타내는 조합으로 결정했으며 각각 표 4, 표 5와 같다. 표 4에 표시된 파라미터들의 정확한 의미는 [4]를 참조하기 바란다. 표 5에서 MCR은 제안하는 방법을 나타낸다. 후보 선택 과정에서 선택된 후보의 수 $n_1$은 각 군집의 크기에 따라 다르게 결정된다. 복원 시오차율은 후보 축소 과정의 $n_2$를 변화시킴으로써 조절하였다.

표 4 초해상도 영상복원 시스템의 파라미터

| 종류 | 파라미터 | 값 |
|---|---|---|
| 훈련 단계 | 패치 크기 | 저주파수 패치: 7×7<br>고주파수 패치: 5×5 |
| | 특징 차원수 | 58 차원 |
| 복원 단계 | 가중치($\alpha$) | 0.75 |
| | 패치 복원간격 | 4 |

표 5 유사 최적 검색 방법 별 파라미터

| 검색 방법 | 파라미터 | 허용오차율 | | |
|---|---|---|---|---|
| | | 1% | 5% | 10% |
| kd-트리 | 탐색 노드 수 | 30 | 20 | 10 |
| BBF | 우선순위 큐 크기 | 15 | 10 | 5 |
| LSH | r | 2000 | 1500 | 1000 |
| | E | 2.5 | 2.5 | 2.5 |
| MCR | 후보선택 군집수(u) | 30 | 30 | 30 |
| | 개략분류 특징차원 (d) | 21 | 21 | 21 |
| | 상세분류 특징차원 (d2) | 31 | 31 | 31 |

### 4.3 실험 결과

실험 결과는 그림 6, 표 6과 같다. 그림 6은 세로 축을 로그 좌표계로 표시하였기 때문에 그래프의 상단에서는 작은 차이도 실제로는 큰 차이를 나타낸다. 영상의 크기가 커질수록 입력 패치의 수뿐만 아니라 패치사전의 크기가 증가하기 때문에 모든 패치 검색 방법에 대하여 복원 시간이 크게 증가하였다. 유사 최적 검색 방법의 경우 허용 오차율이 클수록 속도가 빨랐다. 검색 방법별로는 LSH, BBF, kd-트리의 순으로 빨랐다. 최적 검색의 경우 작은 영상에 대해서는 선형검색이 kd-트리보다 빨랐으나, 큰 영상에 대해서는 kd-트리가 더 빨랐다. 가장 작은 크기인 64×64에서는 최적 검색 방법인 선형 검색이 kd-트리 검색, BBF 검색, LSH보다도 더 빠른 속도를 나타내었다. 이는 알고리즘이 단순하기 때문인 것으로 해석된다. 그러나, 128×128 이상의 영상에서는 유사 최적 검색 방법이 더 빠른 속도를 나타내었으며 영상의 크기가 클수록 그 차이는 급격히 증가하였다.

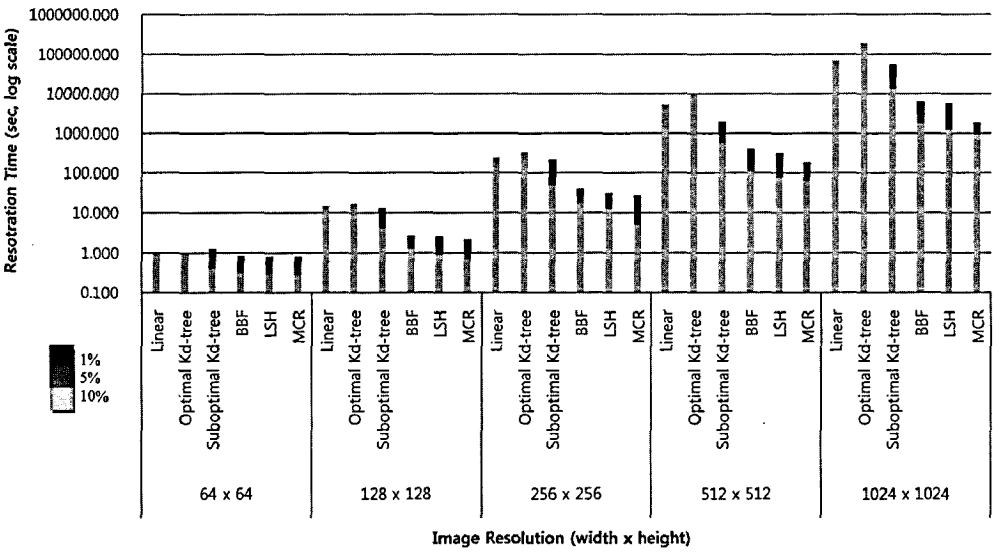



그림 6 패치 검색 방법 별 영상 복원 시간

표 6 해상도, 오차율, 패치 검색 방법 별 영상 복원시간 (초) - 각 해상도 별 10개 영상 샘플의 평균

| 해상도 | 패치사전 크기 | 오차율 (%) | 최적 검색 (오차율0%) | | 유사 최적 검색 | | | | LSH대비 MCR의 속도개선 |
|---|---|---|---|---|---|---|---|---|---|
| | | | 선형 | kd-트리 | kd-트리 | BBF | LSH | MCR | |
| 64 × 64 | 3,249 | 1% | 1.007 | 0.961 | 1.328 | 0.602 | 0.600 | 0.583 | 2.83% |
| | | 5% | | | 0.781 | 0.537 | 0.512 | 0.489 | 4.49% |
| | | 10% | | | 0.413 | 0.319 | 0.301 | 0.281 | 6.64% |
| 128 × 128 | 14,641 | 1% | 15.070 | 17.120 | 9.144 | 2.240 | 2.089 | 1.909 | 8.62% |
| | | 5% | | | 8.059 | 1.762 | 1.451 | 0.986 | 32.05% |
| | | 10% | | | 4.286 | 1.282 | 0.892 | 0.690 | 22.65% |
| 256 × 256 | 62,001 | 1% | 242.125 | 336.242 | 178.084 | 32.338 | 25.87 | 21.692 | 16.15% |
| | | 5% | | | 84.989 | 26.246 | 18.372 | 10.649 | 42.04% |
| | | 10% | | | 49.983 | 17.911 | 12.538 | 5.167 | 58.79% |
| 512 × 512 | 255,025 | 1% | 5343.603 | 10267.870 | 1481.376 | 337.131 | 235.992 | 143.811 | 39.06% |
| | | 5% | | | 1032.444 | 174.905 | 157.415 | 102.828 | 34.68% |
| | | 10% | | | 587.289 | 109.188 | 76.432 | 64.696 | 15.35% |
| 1024 × 1024 | 1,034,289 | 1% | 68469.000 | 185717.986 | 42538.424 | 5195.105 | 4675.595 | 1497.764 | 67.97% |
| | | 5% | | | 24892.690 | 2914.522 | 2331.618 | 1281.024 | 45.06% |
| | | 10% | | | 12884.224 | 1752.382 | 1226.667 | 945.774 | 22.90% |

본 논문에서 제안한 MCR은 LSH보다 더욱 빠른 속도를 보였는데, 영상이 클수록 차이가 더 커졌다. LSH와 비교하였을 때 64 × 64에서는 그 차이가 2.83%에 불과하였으나, 영상이 크고 오차율이 작을수록 그 차이가 증가하였다. 1024 × 1024 영상에 대하여 오차율을 1%로 제한한 경우는 복원 시간이 67.97%나 감소되어 무려 3.12배의 속도차이를 보였다. 따라서, 제안하는 방법이 패치사전의 크기가 크고 허용 오차율이 작을 때 특히 효과적임을 확인할 수 있었다.

## 5. 결 론

예제기반 초해상도 영상복원은 다양한 장점이 있으나 실용적 시스템에 적용하기 위해서는 속도 개선이 요구된다. 효과적인 패치 검색 알고리즘은 전체적인 복원속도 향상을 위해 매우 중요하다. 본 논문에서는 기존에 우수하다고 알려진 다양한 패치 검색 방법론을 실제 초해상도 복원 시스템에 적용하여 성능을 비교 평가하였다. 또한 기존에 초해상도 영상복원에 사용되지 않았던 단계적 후보 축소 방법론을 초해상도 영상복원 시스템에 적용할 것을 제안하였다. 비교 평가 결과 기존의 방법 중에서는 LSH가 가장 좋은 성능을 나타내었으며, 본 논문에서 제안한 단계적 후보 축소 방법의 성능은 LSH보다 더욱 우수하였다.

## 참 고 문 헌


[1] M. E. Tipping and C. M. Bishop. "Bayesian image super-resolution," In S. Thrun S. Becker and K. Obermayer, editors, Advances in Neural Information Processing Systems 15, pp.1279-1286, MIT Press, Cambridge, MA, 2003.

[2] B. Battulga, "Anisotropic Total Variation Method for Text Image Super-Resolution," pp.1-34, Master Thesis, KAIST, 2007.

[3] L. C. Pickup, D. P. Capel, S. J. Roberts, and A. Zisserman, "Bayesian image super-resolution, continued," Advances in Neural Information Processing Systems 19, pp.1089-1096, Cambridge, Mass, USA, December 2006.

[4] William T. Freeman, Thouis R. Jones, Egon C Pasztor, "Example-Based Super-Resolution," *IEEE Computer Graphics and Applications*, vol.22, no.2, pp.56-65, March/April, 2002.

[5] Wang, Q., Tang, X., and Shum, H. "Patch Based Blind Image Super Resolution," In *Proceedings of the Tenth IEEE international Conference on Computer Vision*, 2005.

[6] Simon Baker, Takeo Kanade, "Limits on Super-Resolution and How to Break Them," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol.24, no.9, pp.1167-1183, September, 2002.

[7] J.Park, Y.Kwon, J.Kim, "An Example-Based Prior Model for Text Image Super-resolution," 8th International Conference on Document Analysis and Recognition (ICDAR2005), Seoul, Korea, pp.374-378, 2005.

[8] Kim, Sang-Wook, Charu Aggarwal, Philip Yu, "최근접 질의를 위한 고차원 인덱싱 방법", *한국정보과학회 논문지: 데이타베이스*, vol.28, no.4, pp.632-642, Dec. 2001.

[9] Bentley, J. L., "K-d trees for semidynamic point sets," In *Proceedings of the Sixth Annual Symposium on Computational Geometry, SCG '90. ACM*, New York, NY, 187-197. 1990.

[10] King-ip Lin, H. V. Jagadish, Christos Faloutsos, "The TV-tree: an index structure for high-dimen-

sional data," VLDB, 1994.
[11] Roger Weber, Hans-J. Schek, Stephen Blott, "A Quantitative Analysis and Performance Study for Similarity-Search Methods in High-Dimensional Spaces," *Proc. 24th Int. Conf. VLDB*, pp.194-205, 1998.
[12] Jeffrey S. Beis, David G. Lowe, "Indexing without Invariants in 3D Object Recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol.21, no.10, pp. 1000-1015, Oct., 1999.
[13] Gionis, A., Indyk, P., and Motwani, R, "Similarity Search in High Dimensions via Hashing," In Proceedings of the 25th international Conference on Very Large Data Bases, Sep. 1999.
[14] Ching-Lin Liu, Ryuji Mine, Masashi Koga, "Building Compact Classifier for Large Character Set Recognition Using Discriminative Feature Extraction," icdar, pp.846-850, 2005.
[15] S. C. Ahalt, A. K. Krishnamurthy, P. Chen, and D. E. Melton. Competitive learning algorithms for vector quantization. Neural Networks, 3(3):277-290, 1990.
[16] Z. Wang, A. C. Bovik, H. R. Sheikh and E. P. Simoncelli, "Image quality assessment: From error visibility to structural similarity," *IEEE Transactions on Image Processing*, vol.13, no.4, pp. 600-612, Apr. 2004.

박 규 로
2008년 2월 한동대학교(석사). 2008년 3월~현재 한동대학교 일반대학원 정보통신공학과 석사과정. 관심분야는 영상처리, 패턴인식, 초해상도 영상복원, 저화질 영상 인식

김 인 중
1994년 2월 KAIST 전산학과(학사). 1995년 8월 KAIST 전산학과(석사). 2001년 2월 KAIST 전산학과(박사). 2001년 2월~2006년 2월 (주)인지소프트 책임연구원 2006년 3월~현재 한동대학교 교수. 관심분야는 패턴인식, 영상처리, 인공지능, 모바일 프로그래밍, 증강현실