CHAPTER 10
영상 분할 및 특징 처리

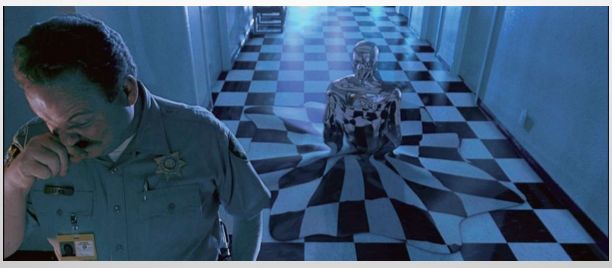
10.4 영상 워핑과 영상 모핑

1

10.4 영상 워핑과 영상 모핑

◆영상 워핑(Warping)

- ❖ 비선형적인 특정한 규칙에 따라 입력 영상을 재추출하여 영상 형태를 변형하는 기술
- ❖ 나사에서 인공위성으로부터 전송된 일그러진 영상을 복원하는 용도로 처음 사용
- ❖ 일반 크기변경과 달리 크기 변화의 정도가 영상 전체에 대해 균일하지 않음
- ❖ 고무판 위에 영상이 있는 것과 같이 임의의 형태로 구부리는 효과→ 고무 시트 변환
- ❖ 사용 예 - 랜즈 왜곡 보정, 스테레오 영상 정합, 파노라마 영상 합성

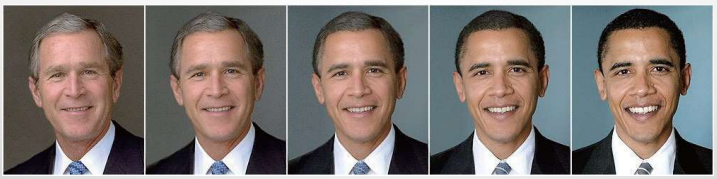


2

10.4 영상 워핑과 영상 모핑

◆영상 모핑(Morphing)

- ❖ 하나의 영상에서 형체가 전혀 다른 영상으로 변하도록 하는 기법
- ❖ 두 개의 서로 다른 영상 사이의 변화하는 과정을 서서히 나타내는 것
- ❖ 변형(metamorphosis)이란 단어에서 유래
- ❖ 조지 루카스가 설립한 특수 효과 전문회사인 ILM(Industrial Light andMagic)이 개발



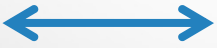
3

10.4 영상 워핑과 영상 모핑

◆예제 - 마우스 드래그로 영상 재배치 하기

$$x' = x + ratio \cdot (pt2.x - pt1.x), \quad ratio = \begin{cases} x < pt1.x & \frac{x}{pt1.x} \\ otherwise & \frac{width - x}{width - pt1.x} \end{cases}$$

$$y' = y$$



4

10.4 영상 워핑과 영상 모핑

예제 10.4.1 마우스 드래그에 반응하는 워핑 변환 – 08.warping.py

```
01  import numpy as np, cv2
02
03  def morphing():                                  # 드래그 거리만큼 영상 왜곡
04      h, w = image.shape[:2]
05      dst = np.zeros((h, w), image.dtype)          # 반환 영상
06      ys = np.arange(0, image.shape[0], 1)         # y 좌표 인덱스
07      xs = np.arange(0, image.shape[1], 0.1)       # x 좌표 인덱스(0.1 간격)
08
09      x1, x10 = pt1[0], pt1[0]*10                  # 0.1 간격의 10배→정수 인덱스 구성
10      ratios = xs / x1                             # 기본 변경 비율
11      ratios[x10:] = (w - xs[x10:]) / (w-x1)       # x 좌표 이상은 다른 비율 적용
12
13      dxs = xs + ratios * (pt2[0] - pt1[0])        # 변경 좌표 인덱스 생성
14      xs, dxs = xs.astype(int), dxs.astype(int)    # 좌표를 정수값으로 변경
15
16      ym, xm  = np.meshgrid(ys, xs)                # 원본 좌표 정방행렬
17      _, dxm = np.meshgrid(ys, dxs)                # 변경 좌표 정방행렬
18      dst[ym, dxm] = image[ym, xm]                 # 원본 좌표를 목적 좌표로 매칭
19      cv2.imshow("image", dst)
20
```

x좌표에 곱해질 비율 계산

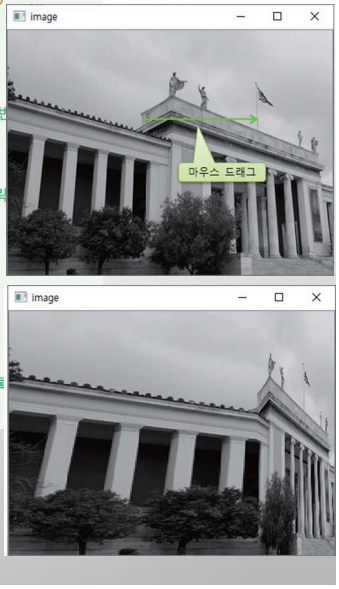
원본 x좌표

변경 x좌표

5

10.4 영상 워핑과 영상 모핑

```
21  def onMouse(event, x, y, flags, param):          # 마우스 이벤트 콜백 함수
22      global pt1, pt2
23      if event == cv2.EVENT_LBUTTONDOWN:           # 마우스 왼쪽버튼 누르기 시 시작좌표 지정
24          pt1 = (x, y)
25      elif event == cv2.EVENT_LBUTTONUP:           # 마우스 왼쪽버튼 떼기 시 종료좌표 지정
26          pt2 = (x, y)                             # 드래그 종료 좌표
27          morphing()                               # 드래그 종료 시 워핑 변
28      elif event == cv2.EVENT_RBUTTONDBLCLK:
29          pt1 = pt2 = (-1, -1)
30          cv2.imshow("image", image)               # 오른쪽 버튼 더블 클릭
31
32  image = cv2.imread("images/warp.jpg", cv2.IMREAD_GRAYSCALE)
33  if image is None: raise Exception("영상파일 읽기 에러")
34
35  pt1 = pt2 = (-1, -1)
36  cv2.imshow("image", image)
37  cv2.setMouseCallback("image", onMouse, 0)        # 마우스 콜백 함수 등록
38  cv2.waitKey(0)
```



6

렌즈 왜곡(lens distortion)

◆렌즈 왜곡 변환 : 변환 행렬로는 구할 수 없는 모양의 변환

❖ 리매핑, 오목 렌즈/볼록 렌즈 왜곡, 방사 왜곡

◆**리매핑(remapping)**

❖ `dst = cv2.remap(src, mapx, mapy, interpolation, dst=None, borderMode=None, borderValue=None)`

- ●규칙성 없이 마음대로 모양을 변환해 주는 함수
- ●mapx, mapy : x축과 y축으로 이동할 좌표, src와 동일한 크기, float32
  - ✓ mapx[0][0]=10, mapy[0][0]=5 : src 좌표 (0,0) 픽셀을 dst 좌표 (10,5)로 옮기라는 의미
  - ✓ 초기화
    - `mapy, mapx = np.indices( (rows, cols), dtype=np.float32)` (rows, cols) 크기의 행렬을 생성하여 자신의 인덱스를 값으로 초기화함
- ●재배치할 좌표가 정수로 떨어지지 않는 경우엔 자동으로 보정함

7

렌즈 왜곡(lens distortion)

❖ 예제 : 삼각함수를 이용한 비선형 remapping

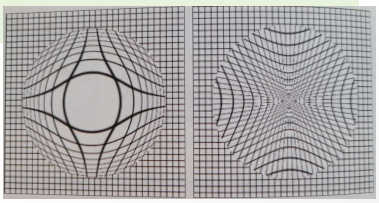
- ●x좌표, y좌표에 sin, cos 함수를 적용하여 새로운 사인파 코사인파 곡선으로 왜곡된 이미지를 만듬

```
l = 20      # 파장(wave length)
amp = 15    # 진폭(amplitude)

img = cv2.imread('../img/taekwonv1.jpg')
rows, cols = img.shape[:2]

# 초기 매핑 배열 생성 ---①
mapy, mapx = np.indices((rows, cols),dtype=np.float32)

# sin, cos 함수를 적용한 변형 매핑 연산 ---②
sinx = mapx + amp * np.sin(mapy/l)
cosy = mapy + amp * np.cos(mapx/l)

# 영상 매핑 ---③
img_sinx=cv2.remap(img, sinx, mapy, cv2.INTER_LINEAR) # x축만 sin 곡선 적용
img_cosy=cv2.remap(img, mapx, cosy, cv2.INTER_LINEAR) # y축만 cos 곡선 적용
# x,y 축 모두 sin, cos 곡선 적용 및 외곽 영역 보정
img_both=cv2.remap(img, sinx, cosy, cv2.INTER_LINEAR,
\                  None, cv2.BORDER_REPLICATE)

# 결과 출력
cv2.imshow('origin', img)
cv2.imshow('sin x', img_sinx)
cv2.imshow('cos y', img_cosy)
cv2.imshow('sin cos', img_both)
```

영상 테두리(마지막) 값을 그대로 사용

8

렌즈 왜곡

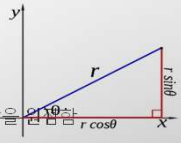


◆**오목 렌즈와 볼록 렌즈 왜곡**

❖ 원의 중심에서의 거리를 증가시키는 변환이 볼록렌즈, 줄이는 변환이 오목렌즈 효과를 줌

❖ x축과 y축 형태의 좌표 시스템을 **직교 좌표계(Cartesian coordinate system)**를 원점으로부터의 거리(r)와 사잇각(Θ)을 이용해서 (r, Θ)로 나타내는 **극좌표계(Polar coordinate system)로 변환**이 필요함

- `r, theta = cv2.cartToPolar(x, y)`: 직교 좌표 → 극좌표 변환
- `x, y = cv2.polarToCart(r, theta)`: 극좌표 → 직교 좌표 변환



❖ 좌표의 기준점 변환도 중요

- 일반적으로 직교 좌표계는 좌측 상단을 원점(0, 0)으로 정함 -> 극좌표는 이미지의 중앙을 원점함
- 이미지의 중앙을 (0, 0)으로 두기 위해서 좌표의 값을 -1 ~ 1로 정규화해야 함.
  - ✓ `mapx = 2*mapx/(cols-1)-1, mapy = 2*mapy/(rows-1)-1`
- 이후에 다시 좌상단 기준점으로 변경

❖ exp는 이미지의 왜곡 지수를 나타내는 변수

- 1이면 원본과 동일하게 하고, 1보다 작으면 오목 렌즈 효과를 내고, 1보다 크면 볼록 렌즈 효과를 냄

❖ scale은 이미지에서 렌즈 효과를 주고 싶은 원 모양 영역의 크기를 비율로 나타낸 것

- scale=1은 100%를 의미

9

렌즈 왜곡

```
img = cv2.imread('../img/taekwonv1.jpg')
rows, cols = img.shape[:2]

# ---① 설정 값 셋팅
exp = 2       # 볼록, 오목 지수 (오목 : 0.1 ~ 1, 볼록 : 1.1~)
scale = 1           # 변환 영역 크기 (0 ~ 1)

# 매핑 배열 생성 ---②
mapy, mapx = np.indices((rows, cols),dtype=np.float32)

# 좌상단 기준좌표에서 -1~1로 정규화된 중심점 기준 좌표로 변경 ---③
mapx = 2*mapx/(cols-1)-1
mapy = 2*mapy/(rows-1)-1

# 직교좌표를 극 좌표로 변환 ---④
r, theta = cv2.cartToPolar(mapx, mapy)

# 왜곡 영역만 중심확대/축소 지수 적용 ---⑤
r[r< scale] = r[r<scale] **exp

# 극 좌표를 직교좌표로 변환 ---⑥
mapx, mapy = cv2.polarToCart(r, theta)

# 중심점 기준에서 좌상단 기준으로 변경 ---⑦
mapx = ((mapx + 1)*cols-1)/2
mapy = ((mapy + 1)*rows-1)/2

# 재매핑 변환
distorted = cv2.remap(img,mapx,mapy,cv2.INTER_LINEAR)

cv2.imshow('origin', img)
cv2.imshow('distorted', distorted)
```

10

렌즈 왜곡

◆**방사 왜곡**

❖ 카메라 렌즈는 동그랗고 영상은 사각형 -> 렌즈의 가장자리에서 왜곡 발생 -> Barrel 왜곡

●해결책 : $r_d = r_u(1 + k_1 r_u^2 + k_2 r_u^4 + k_3 r_u^6)$

✓ r_u는 왜곡 변형 전, r_d는 왜곡 변형 후, k_1, k_2, k_3는 왜곡 계수

```
# 중앙점 좌표로 -1~1 정규화 및 극좌표 변환 ---③
mapx = 2*mapx/(cols-1)-1
mapy = 2*mapy/(rows-1)-1
r, theta = cv2.cartToPolar(mapx, mapy)

# 방사 왜곡 변영 연산 ---④
ru = r*(1+k1*(r**2) + k2*(r**4) + k3*(r**6))

# 직교좌표 및 좌상단 기준으로 복원 ---⑤
mapx, mapy = cv2.polarToCart(ru, theta)
mapx = ((mapx + 1)*cols-1)/2
mapy = ((mapy + 1)*rows-1)/2

# 리매핑 ---⑥
distored = cv2.remap(img,mapx,mapy,cv2.INTER_LINEAR)
```

11

단원 요약

◆영상 워핑은 영상에서 비선형적인 특정한 규칙에 따라 입력 영상을 재추출(resampling)하여 영상의 형태를 변형시키는 기술이다. 영상을 다른 여러 방향으로 늘이거나 크기를조절하는 기법으로 수순한 스케일링과 달리 크기 변화의 정도가 영상 전체에 대해 균일하지 않는 것이 특징이다.

◆영상 모핑(Morping)은 조지 루카스가 설립한 특수 효과 전문회사인 ILM(Industrial Light and Magic)이 개발한 기법으로 변형(metamorphosis)이란 단어에서 유래되었다. 이것은 하나의 영상에서 형체가 전혀 다른 영상으로 변하도록 하는 기법을 말한다

12