IMA 4509

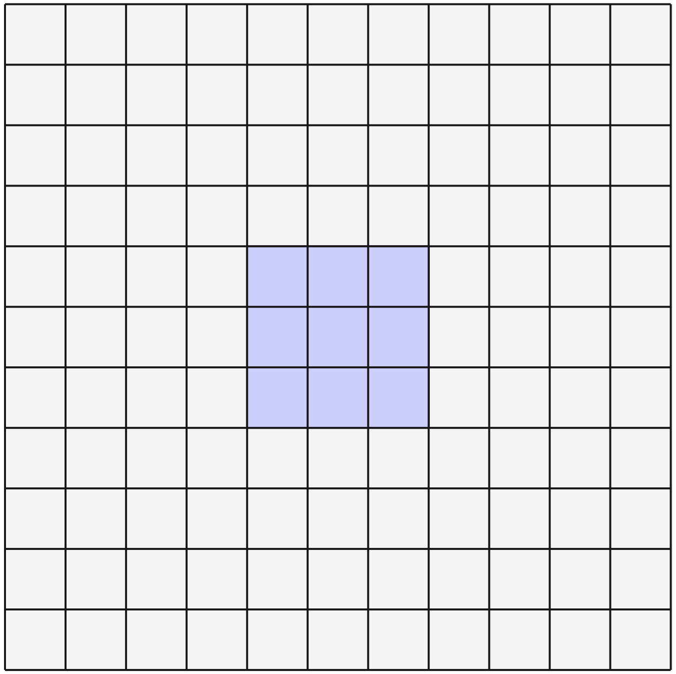
Visual content analysis

Programming techniques for image analysis: Neighborhood operations

Nicolas ROUGON
ARTEMIS Department

Neighborhood operations

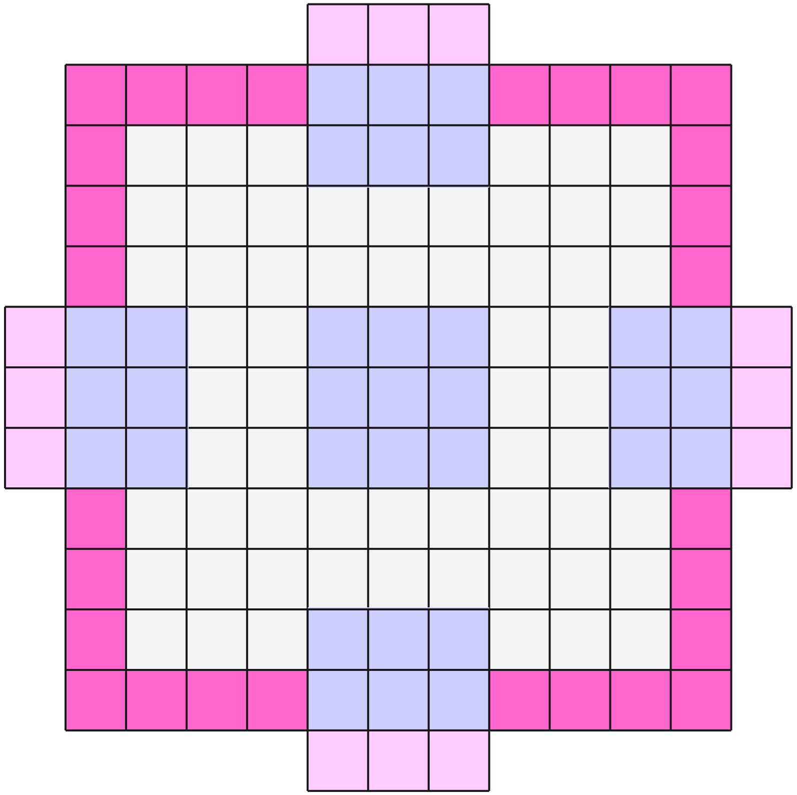
Computations in the pixel neighborhood



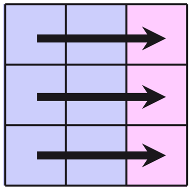
- Linear filtering
  - > smoothing
  - > differential filtering
- Local statistics
  - > statistical filtering
- Order / rank statistics
  - > rank-filtering
  - > morphological transforms
- …

Neighborhood operations

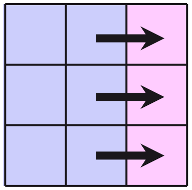
Generic case & boundary conditions



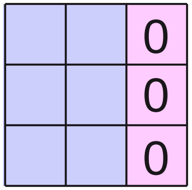
- Mirroring
  - > Neumann-type



- Replication
  - > Dirichlet-type



- Zero-padding
  - > Dirichlet-type



Neighborhood operations

Challenges

- Managing image core & borders in a unified way
  > compact code

Solutions

- Neighborhood encoding
  > data structure-dependent

Challenges

- Minimizing the number of operations / pixel
  > computational efficiency

Solutions

- Pre-compute all quantities which remain constant during raster scan

Neighborhood encoding

- **Images as 2D arrays**
  - Addressing neighbors
    - > "cardinal points" indexing

L[j+n][i+w]	L[j+n][i]	L[j+n][i+e]
L[j][i+w]	L[j][i]	L[j][i+e]
L[j+s][i+w]	L[j+s][i]	L[j+s][i+e]

  - Neighborhood encoding
    - > offset matrix

-1,-1	-1,0	-1,1
0,-1	0,0	0,1
1, -1	1,0	1,1

  - Boundary conditions are enforced by updating the offset matrix

Neighborhood encoding

Images as 1D arrays

- Addressing neighbors
  - > “cardinal points” indexing

L[i+n+w]	L[i+n]	L[i+n+e]
L[i+w]	L[i]	L[i+e]
L[i+s+w]	L[i+s]	L[i+s+e]

- Neighborhood encoding
  - > offset matrix

-xsize-1	-xsize	-xsize+1
-1	0	1
xsize-1	xsize	xsize+1

- Boundary conditions are enforced by updating the offset matrix

Neighborhood processing

- **Example #1 - Linear filtering**
  - Images as 2D arrays

```
xsize_1 = xsize – 1;                          // pre-compute constants
ysize_1 = ysize – 1;
for (j=0; j<ysize; j++) {                     // raster scan along rows
   n = ((j==0) ? 1 : -1);                     // row index offsets
   s = ((j==ysize_1) ? -1 : 1);
   jn = j + n; js = j + s;                    // pre-compute row indices
   for (i=0; i<xsize); i++) {                 // raster scan along columns
          w = ((i==0) ? 1 : -1);              // column index offsets
          e = ((i==xsize_1) ? -1 : 1);
           iw = i + w;  ie = i + e;           // pre-compute col. indices
```

Neighborhood processing

Example #1 - Linear filtering

- Images as 2D arrays (cont'd)

```
        // neighborhood operation
        out[j][i] = a11*L[jn][iw] + a12*L[jn][i] + a13*L[jn][ie] +
                    a21*L[j][iw] + a22*L[j][i] + a23*L[j][ie] +
                    a31*L[js][iw] + a32*L[js][i] + a33*L[js][ie];
    }
}
```

Neighborhood processing

- **Example #1 - Linear filtering**
  - Images as 1D arrays

```
p_L = L;                                        // initialize pointers
p_out = out;
for (j=0; j<ysize; j++) {                       // raster scan along rows
   n = ((j==0) ? xsize : -xsize);               // line index offsets
   s = ((j==ysize_1) ? -xsize : xsize);
   for (i=0; i<xsize); i++) {                   // raster scan along columns
         w = ((i==0) ? 1 : -1);                 // column index offsets
         e = ((i==xsize_1) ? -1 : 1);
```

Neighborhood processing

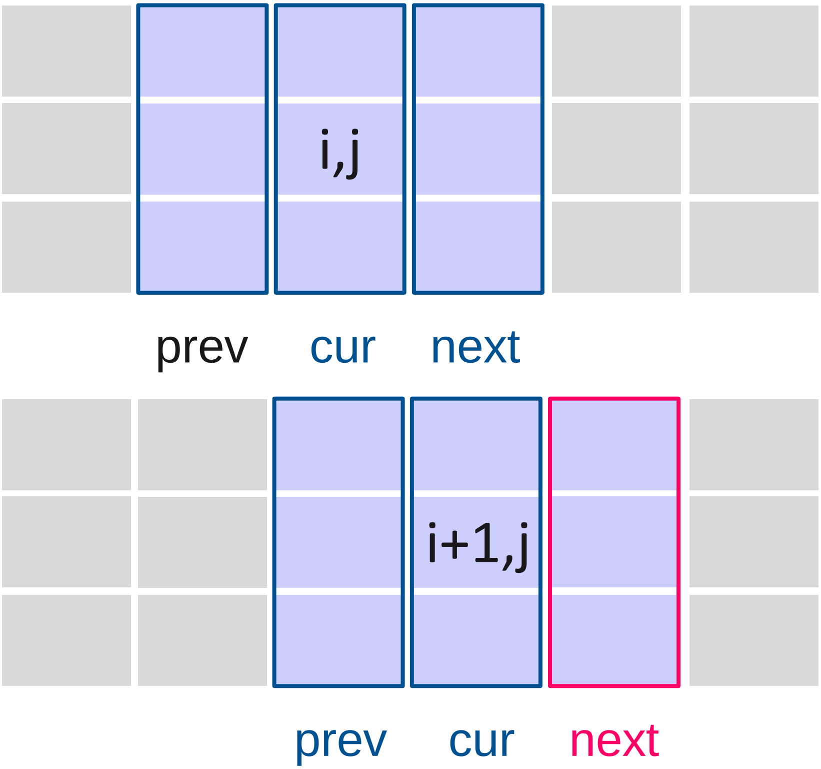
- **Example #1 - Linear filtering**
  - Images as 1D arrays (cont'd)

```
            // neighborhood operation
            *p_out =
                *(p_L+n+w)*a11 + *(p_L+n)*a12+ *(p_L+n+e)*a13 +
                *(p_L+w)*a21 + *p_L*a22 + *(p_L+e)*a23 +
                *(p_L+s+w)*a31 + *(p_L+s)*a32 + *(p_L+s+e)*a33;

            // move to next pixel
            p_L++; p_out++;
    }
}
```

Neighborhood processing

Example #2 - Average filter



- Neighborhood average decomposes as a sum of columns averages

- At the next pixel, a single column average needs to be computed, the others being already available
  - > sliding window implementation

Neighborhood processing

- **Example #2 - Average filter**
  - Images as 2D arrays

```
for (j=0; j<ysize; j++) {                          // raster scan along rows
   n = ((j==0) ? 1 : -1);                          // row index offsets
   s = ((j==ysize_1) ? -1 : 1);
   jn = j + n; js = j + s;                         // pre-compute row indices
   current = L[jn][1] + L[j][1] + L[js][1];        // line start initialization
   next    = L[jn][0] + L[j][0] + L[js][0];
   for (i=0; i<xsize); i++) {                      // raster scan along columns
         e = ((i==xsize_1) ? -1 : 1);              // column index offsets
         ie = i + e;                               // pre-compute col. indices
```

Neighborhood processing

Example #2 - Average filter

Images as 2D arrays (cont'd)

```
            // rotating buffer
            previous = current;
            current = next;
            next = L[jn][ie]  + L[j][ie] +  L[js][ie];

            // neighborhood operation
            out[j][i] = (previous + current + next) / 9.0;
    }
}
```

Neighborhood processing

Example #2 - Average filter

Images as 1D arrays

```
p_L = L;                                    // initialize image pointers
p_out = out;
for (j=0; j<ysize; j++) {                   // raster scan along rows
  n = ((j==0) ? 1 : -1);                    // row index offsets
  s = ((j==ysize_1) ? -1 : 1);
                                            // line start initialization
  current = *(p_L+n+1) + *(p_L+1) + *(p_L+s+1);
  next = *(p_L+n) + *p_L + *(p_L+s);
  for (i=0; i<xsize); i++) {                // raster scan along columns
       e = ((i==xsize_1) ? -1 : 1);         // column index offset
```

Neighborhood processing

- **Example #2 - Average filter**
  - Images as 1D arrays (cont'd)

```
            // rotating buffer
            previous = current;
            current = next;
            next = *(p_L+n+e)  + *(p_L+e) +  *(p_L+s+e);

            // neighborhood operation
            *p_out = (previous + current + next) / 9.0;

            // move pointers to next pixel
             p_L++; p_out++;
    }
  }
```

IMA 4509

Visual content analysis

Programming techniques for image analysis: Neighborhood operations

Nicolas ROUGON

ARTEMIS Department