POLITECHNIKA GDAŃSKA

Gesture based interface

Tomasz Kocejko

Touch UI

- Touch interface is where the users can interact with machine using the sense of touch.
- Users can use their hands or a stylus to touch the screen when making selections on the machine
- This interface much relies on graphical user interface (GUI) which allows user to see what (icons) they should touch when making a selections
- In some cases of visual impairments , there is a level of interaction based upon tactile or Braille input.

POLITECHNIKA GDAŃSKA

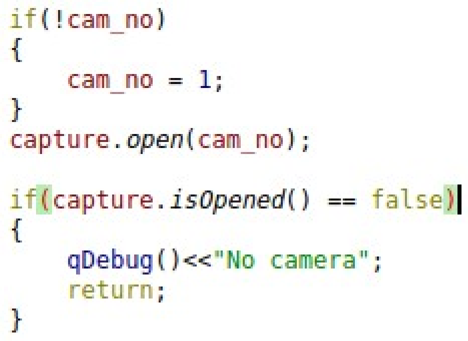
Gesture based UI

- This interface is where computers or machine can interpret human gestures via mathematical system.
- The gestures can originate from human body especially our face or hands.
- Many approaches were made by using cameras and the computer algorithms to interpret the gestures.
- Air-based gesture is a begin for computers to understand human body language.

Basic algorithm

Image acquisition

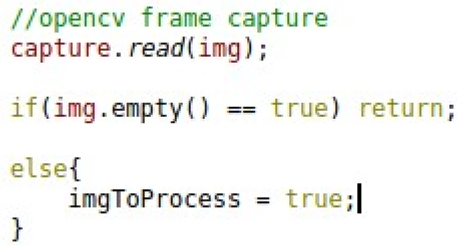
Set and open device supported by opencv library...



```
if(!cam_no)
{
    cam_no = 1;
}
capture.open(cam_no);

if(capture.isOpened() == false)
{
    qDebug()<<"No camera";
    return;
}
```

… retrieve frame and forward for further processing



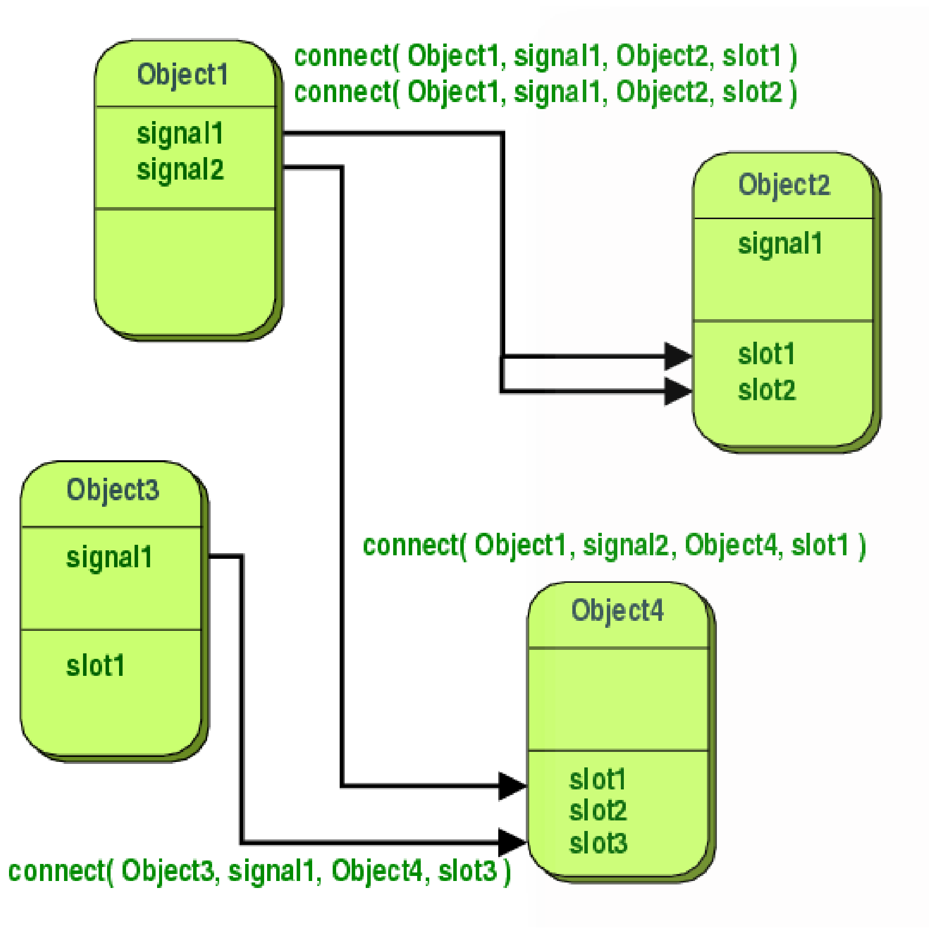
```
//opencv frame capture
capture.read(img);

if(img.empty() == true) return;

else{
    imgToProcess = true;
}
```

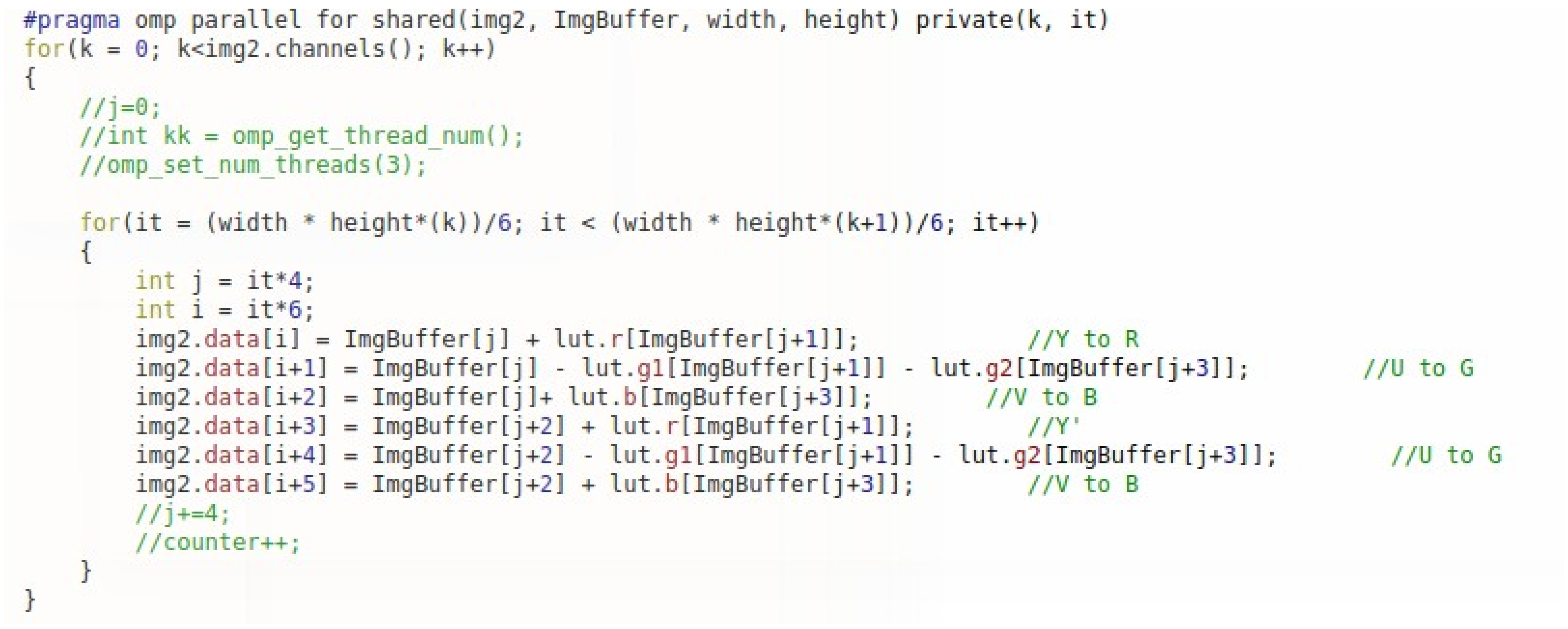
POLITECHNIKA GDAŃSKA

Signal-Slot connectio


Object1
signal1
signal2
connect(Object1, signal1, Object2, slot1)
connect(Object1, signal1, Object2, slot2)
Object2
signal1
slot1
slot2
Object3
signal1
slot1
connect(Object1, signal2, Object4, slot1)
Object4
slot1
slot2
slot3
connect(Object3, signal1, Object4, slot3)

Multithreading

Implementation of OMP library to speedup image color conversion

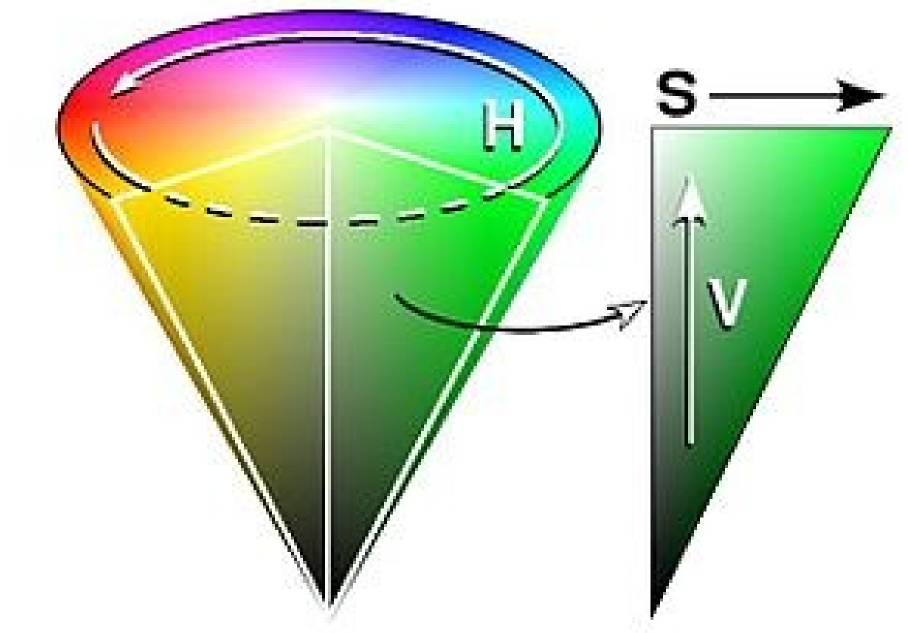


```
#pragma omp parallel for shared(img2, ImgBuffer, width, height) private(k, it)
for(k = 0; k<img2.channels(); k++)
{
    //j=0;
    //int kk = omp_get_thread_num();
    //omp_set_num_threads(3);

    for(it = (width * height*(k))/6; it < (width * height*(k+1))/6; it++)
    {
        int j = it*4;
        int i = it*6;
        img2.data[i] = ImgBuffer[j] + lut.r[ImgBuffer[j+1]];            //Y to R
        img2.data[i+1] = ImgBuffer[j] - lut.g1[ImgBuffer[j+1]] - lut.g2[ImgBuffer[j+3]];        //U to G
        img2.data[i+2] = ImgBuffer[j]+ lut.b[ImgBuffer[j+3]];        //V to B
        img2.data[i+3] = ImgBuffer[j+2] + lut.r[ImgBuffer[j+1]];        //Y'
        img2.data[i+4] = ImgBuffer[j+2] - lut.g1[ImgBuffer[j+1]] - lut.g2[ImgBuffer[j+3]];        //U to G
        img2.data[i+5] = ImgBuffer[j+2] + lut.b[ImgBuffer[j+3]];        //V to B
        //j+=4;
        //counter++;
    }
}
```

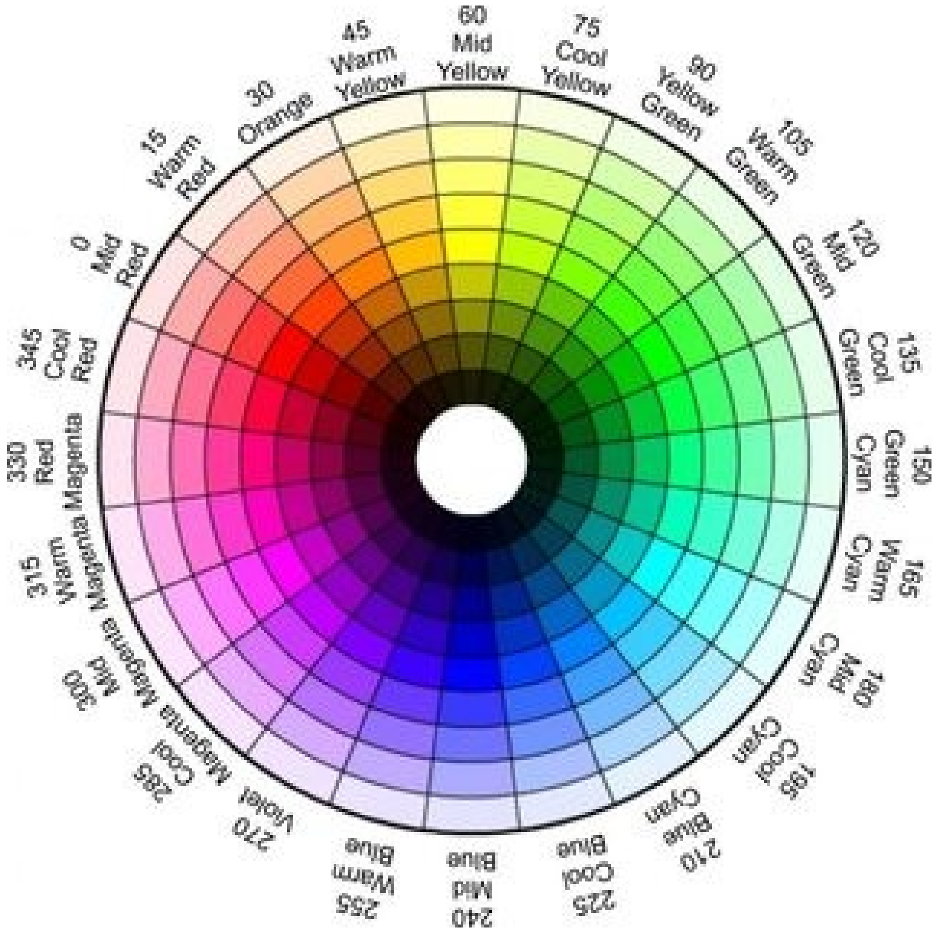
HSV

- most common cylindrical-coordinate representations of points in an RGB color model
- In each cylinder, the angle around the central vertical axis corresponds to "hue", the distance from the axis corresponds to "saturation", and the distance along the axis corresponds to "lightness", "value" or "brightness".



POLITECHNIKA GDAŃSKA

HSV color wheel



POLITECHNIKA GDAŃSKA

Morphology operations - Dilation

- Suppose that X is the set of Euclidean coordinates corresponding to the input binary image, and that K is the set of coordinates for the structuring element.
- Let Kx denote the translation of K so that its origin is at x.
- Then the dilation of X by K is simply the set of all points x such that the intersection of Kx with X is non-empty

Morphology operations - Dilation

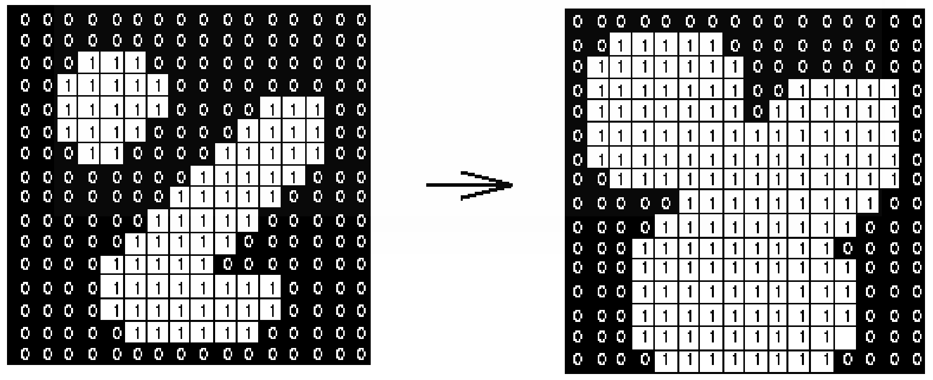
1	1	1
1	1	1
1	1	1

Set of coordinate points =

{ (-1, -1), (0, -1), (1, -1),

(-1, 0), (0, 0), (1, 0),

(-1, 1), (0, 1), (1, 1) }



0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0
0	0	1	1	1	1	1	0	0	0	0	0	0	0	0	0
0	0	1	1	1	1	1	0	0	0	0	1	1	1	0	0
0	0	1	1	1	1	0	0	0	0	1	1	1	1	0	0
0	0	0	1	1	0	0	0	0	1	1	1	1	1	0	0
0	0	0	0	0	0	0	0	1	1	1	1	1	0	0	0
0	0	0	0	0	0	0	1	1	1	1	1	0	0	0	0
0	0	0	0	0	0	1	1	1	1	1	0	0	0	0	0
0	0	0	0	0	1	1	1	1	1	0	0	0	0	0	0
0	0	0	0	1	1	1	1	1	0	0	0	0	0	0	0
0	0	0	0	1	1	1	1	1	1	1	1	0	0	0	0
0	0	0	0	1	1	1	1	1	1	1	1	0	0	0	0
0	0	0	0	0	1	1	1	1	1	1	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

$\longrightarrow$

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	1	1	1	1	1	0	0	0	0	0	0	0	0	0
0	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0
0	1	1	1	1	1	1	1	0	0	1	1	1	1	1	0
0	1	1	1	1	1	1	1	0	1	1	1	1	1	1	0
0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0
0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0
0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	0
0	0	0	0	0	1	1	1	1	1	1	1	1	1	0	0
0	0	0	0	1	1	1	1	1	1	1	1	1	0	0	0
0	0	0	1	1	1	1	1	1	1	1	1	0	0	0	0
0	0	0	1	1	1	1	1	1	1	1	1	1	0	0	0
0	0	0	1	1	1	1	1	1	1	1	1	1	0	0	0
0	0	0	1	1	1	1	1	1	1	1	1	1	0	0	0
0	0	0	1	1	1	1	1	1	1	1	1		0	0	0
0	0	0	0	1	1	1	1	1	1	1	1	0	0	0	0

POLITECHNIKA GDAŃSKA

Morphology operations - Dilation

- Suppose that X is the set of Euclidean coordinates corresponding to the input binary image, and that K is the set of coordinates for the structuring element.
- Let Kx denote the translation of K so that its origin is at x.
- Then the erosion of X by K is simply the set of all points x such that Kx is a subset of X.

Morphology operations - Dilation

1	1	1
1	1	1
1	1	1

Set of coordinate points =

{ (-1, -1), (0, -1), (1, -1),

(-1, 0), (0, 0), (1, 0),

(-1, 1), (0, 1), (1, 1) }

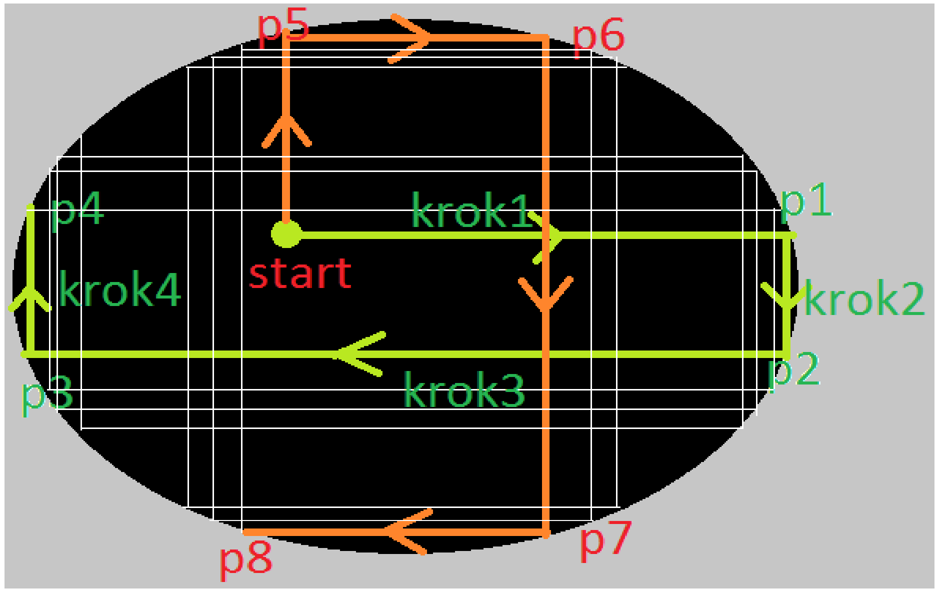
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0
0	0	1	1	1	1	1	0	0	0	0	0	0	0	0	0
0	0	1	1	1	1	1	0	0	0	0	1	1	1	0	0
0	0	1	1	1	1	0	0	0	0	1	1	1	1	0	0
0	0	0	1	1	0	0	0	0	1	1	1	1	1	0	0
0	0	0	0	0	0	0	0	1	1	1	1	1	0	0	0
0	0	0	0	0	0	0	1	1	1	1	1	0	0	0	0
0	0	0	0	0	0	1	1	1	1	1	0	0	0	0	0
0	0	0	0	0	1	1	1	1	1	0	0	0	0	0	0
0	0	0	0	1	1	1	1	1	0	0	0	0	0	0	0
0	0	0	0	1	1	1	1	1	1	1	1	0	0	0	0
0	0	0	0	1	1	1	1	1	1	1	1	0	0	0	0
0	0	0	0	0	1	1	1	1	1	1	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

$\longrightarrow$

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0
0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0
0	0	0	0	0	0	1	1	1	1	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

POLITECHNIKA GDAŃSKA

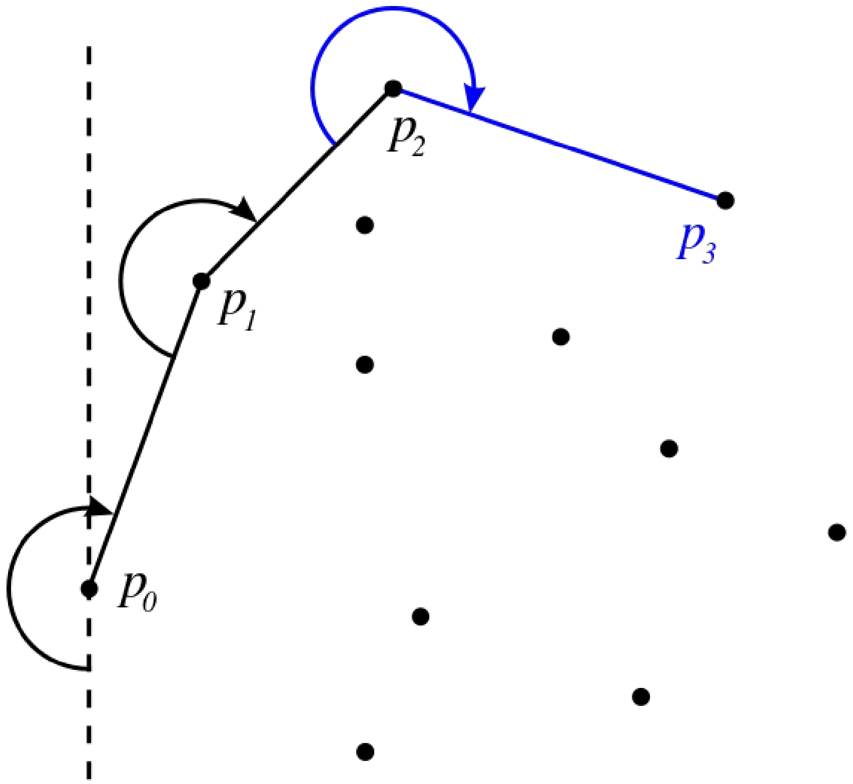
Contour detection



- Ransac
- Bug follower
- Combination

POLITECHNIKA GDAŃSKA

Convex hull „gift wraping”

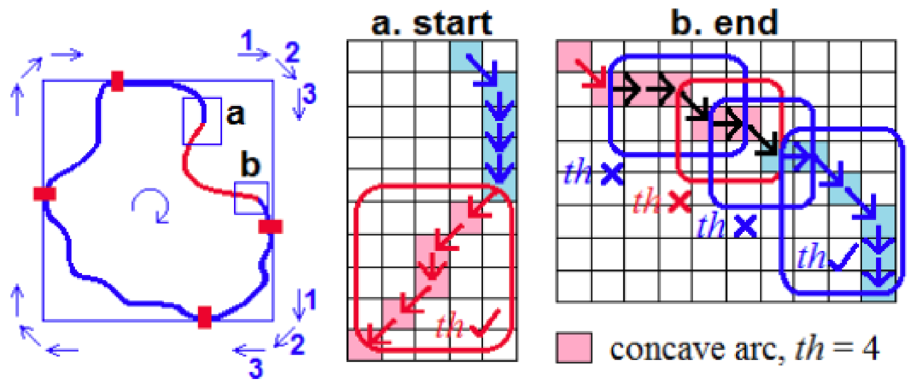


```
jarvis(S)
   pointOnHull = leftmost point in S
   i = 0
   repeat
      P[i] = pointOnHull
      endpoint = S[0]         // initial endpoint for a candidate edge on the hull
      for j from 1 to |S|
         if (endpoint == pointOnHull) or (S[j] is on left of line from P[i] to endpoint)
            endpoint = S[j]   // found greater left turn, update endpoint
      i = i+1
      pointOnHull = endpoint
   until endpoint == P[0]      // wrapped around to first hull point
```

POLITECHNIKA GDAŃSKA

Convexity detection algorithm

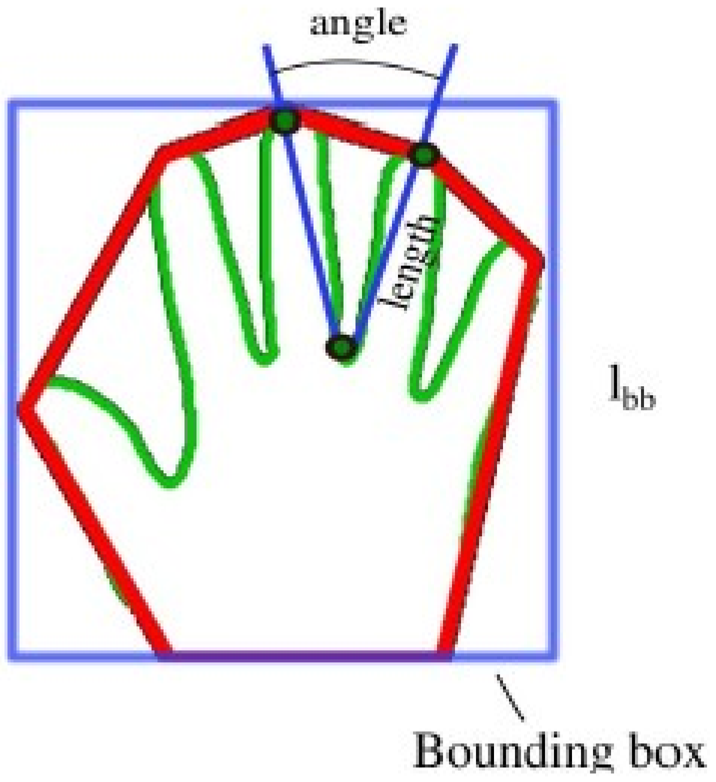
- Input:
- Superpixel contour s (clock-wise), concave threshold th
- Steps:
- 1.Smooth s using B-spline curve algorithm.
- 2. Draw bounding-box and split s into four sections.
- 3. ret = Φ.
- 4. for each section do
- 4.1. Determine three main directions in sequence.
- 4.2. Find starting points of concave arc when pixel movements violate main direction for more than th steps.
- 4.3. Find ending points of concave arc when pixel movements accord with main direction for more than th steps after string points detected.
- 4.4. arc = combination of all the pieces in this section.
- 4.5. ret = ret ∪ arc
- 5. retuen ret



*Convexity detection with application in saliency and attention selection
http://homes.cs.washington.edu/~luyao/conv.html

Defects filtration example

- length<0.4lbb
- angle>80



Improvements

Gesture clasification methods

- Hidden Markov Model
- haar-like features
- Finit state machines
- Neural networks
- ...

*Gesture Recognition: A Survay

POLITECHNIKA GDAŃSKA

Tomorrow is today