Pattern Recognition Systems – Lab 2

RANSAC – fitting a circle to a set of points

1. Objectives

The purpose of this laboratory session is to apply RANSAC, the algorithm introduced in the previous lab, to a more complex problem: estimating a circle to fit to a set of 2-dimensional points.

2. Theoretical Background

The RANSAC algorithm is summarized below [3]:

Objective: Robust fit of a model to a data set S which contains outliers.

Algorithm:

1. Randomly select a sample of s data points from S and instantiate the model from this subset.
2. Determine the set of data points S_i which is within a distance threshold t of the model. The set S_i, is the consensus set of the sample and defines the inliers of S.
3. If the size of S_i (the number of inliers) is greater than some threshold T, re-estimate the model using all the points in S_i and terminate.
4. If the size of S_i is less than T, select a new subset and repeat the above.
5. After N trials the largest consensus set S_i is selected, and the model is re-estimated using all the points in the subset S_i.

3. RANSAC for fitting a circle to a set of points

The problem we need to solve is that being given a set of 2D data points, find the circle which minimizes the sum of the squared distances of the contour points to the circle:

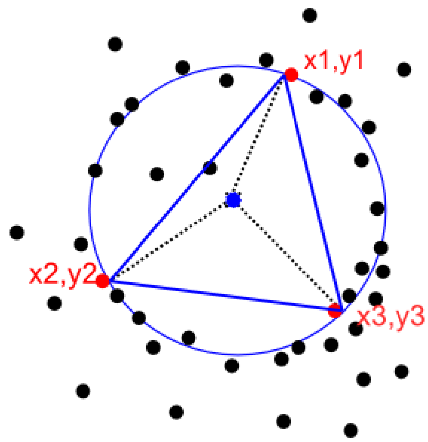
$\varepsilon^2 = \sum_{i=1}^{n} \left(\sqrt{(x_i - x_c)^2 + (y_i - y_c)^2} - r\right)^2$ subject to the condition that none of the valid points deviates from the circle by more than t units. Here (x_c, y_c) represent the coordinates of the center of the circle and r is its radius.

This is actually two problems: a circle fit to the data; and a classification of the data into inliers (valid points) and outliers. The threshold t is set according to the measurement noise.

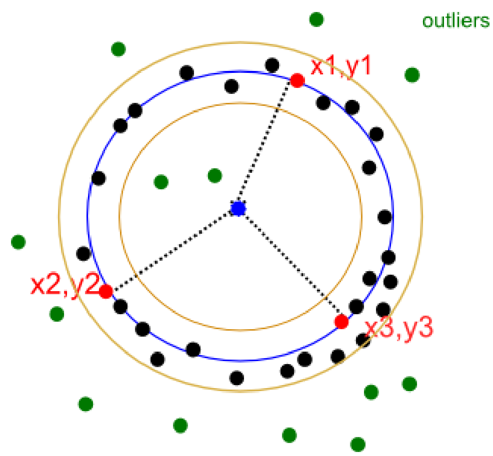
Three of the points are selected randomly; these points define a circle. The *support* for the circle is measured by the number of points that lie within a distance threshold. This random selection is repeated a number of times and the circle with most support is deemed the robust fit. The points within the threshold distance are the inliers (and constitute the eponymous *consensus* set). The intuition is that if one of the points is an outlier then the circle will not gain much support. Furthermore, scoring a circle by its support has the additional advantage of favoring better fits.

The algorithm (slightly simplified version of RANSAC):

1. Given a set of points, select randomly three points ($s=3$). Find the center and the radius of the circumscribed circle to the triangle formed by the three points.



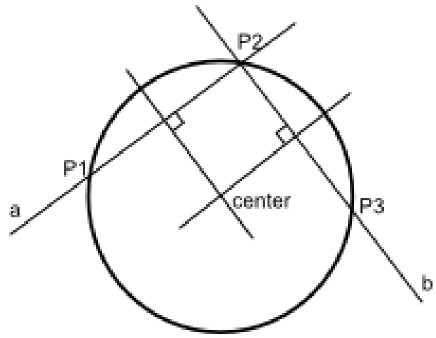
2. Compute the distance from all the other points to the circle and find the inliers and outliers. Compute the size of the consensus set S_i. Optionally, the algorithm can stop if the size of S_i is greater than T (see Lab 1).



3. After at most N trials the largest consensus set S_i is selected and the model associated to S_i is considered as solution.

4. Mathematical background

4.1. Given three points P_1, P_2, and P_3 on a plane, find the center and the radius of the circle passing through the three points



The coordinates of the center (x_c, y_c) can be computed as follows:

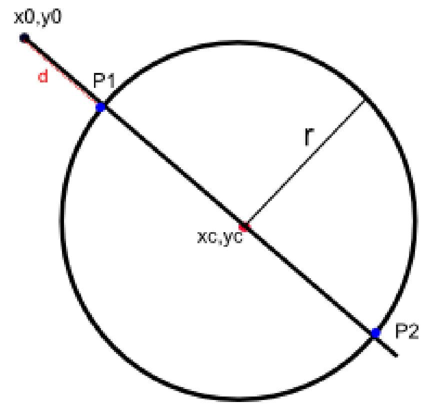
- Two lines can be formed through 2 pairs of the three points, the first (*a*) passes through the first two points P_1 and P_2. Line *(b)* passes through the next two points P_2 and P_3.
- The equation of these two lines is: $y_a = m_a(x - x_1) + y_1$ and $y_b = m_b(x - x_2) + y_2$, where m is the slope of the line given by: $m_a = \frac{y_2 - y_1}{x_2 - x_1}$ and $m_b = \frac{y_3 - y_2}{x_3 - x_2}$
- The centre of the circle is the intersection of the two lines perpendicular to and passing through the midpoints of the lines P_1P_2 and P_2P_3.
- The perpendicular of a line with slope *m* has slope *-1/m*, thus equations of the lines perpendicular to lines *a* and *b* and passing through the midpoints of P_1P_2 and P_2P_3 are:

$$y_a' = -\frac{1}{m_a}\left(x - \frac{x_1 + x_2}{2}\right) + \frac{y_1 + y_2}{2}$$
$$y_b' = -\frac{1}{m_b}\left(x - \frac{x_2 + x_3}{2}\right) + \frac{y_2 + y_3}{2}$$

- These two lines intersect at the center of the circle, hence solving the above equations for *x* gives: $x_c = \frac{m_a m_b (y_1 - y_3) + m_b(x_1 + x_2) - m_a(x_2 + x_3)}{2(m_b - m_a)}$
- Using the value obtained for *x* we can compute *y* by substituting the *x* value into one of the equations of the perpendiculars, hence, taking the first equation we obtain: $y_c = -\frac{1}{m_a}\left(x_c - \frac{x_1 + x_2}{2}\right) + \frac{y_1 + y_2}{2}$

The radius of the circle can be computed considering that the point P_1 lies on the circle, and having the coordinates of the center *(x_c, y_c)*, the radius is the distance between the center and the point P_1: $r = \sqrt{(x_c - x_1)^2 + (y_c - y_1)^2}$

4.2. Given a circle and a point (x_0, y_0), determine the distance from the point to the circle



Define the line determined by the point (x_0, y_0) and the center of the circle. Let P_1 and P_2 be the intersections of the line with the circle.
The distance between the point and the circle is given by:

$$d = \left| \sqrt{(x_c - x_0)^2 + (y_c - y_0)^2} - r \right|$$

5. Implementing RANSAC for circles in DIBLook

```
void CDibView::OnProcessingRANSACCircle()
{
      BEGIN_PROCESSING();
      /*************************************************
      TODO:
       Write down the code for RANSAC circle
      *************************************************/

      /* Drawing a circle on the image */
      CDC dc;
      dc.CreateCompatibleDC(0);
      CBitmap ddBitmap;
      HBITMAP hDDBitmap = CreateDIBitmap(::GetDC(0),
                  &((LPBITMAPINFO)lpS)->bmiHeader, CBM_INIT, lpSrc,
                  (LPBITMAPINFO)lpS, DIB_RGB_COLORS);

      ddBitmap.Attach(hDDBitmap);
      CBitmap* pTempBmp = dc.SelectObject(&ddBitmap);
      CPen pen(PS_SOLID, 1, RGB(255,0,0));
      CPen *pTempPen = dc.SelectObject(&pen);

      /* draw a circle having radius r and center a point of
      coordinates (x,y)*/
      int x = 100;
      int y = 90;
      int r = 20;
```

```
    dc.MoveTo ( (int)(x + r), dwHeight-1-y );
    dc.AngleArc(x, dwHeight-1-y, r, 0, 360);

    dc.SelectObject(pTempPen);
    dc.SelectObject(pTempBmp);
    GetDIBits(dc.m_hDC, ddBitmap, 0, dwHeight, lpDst,
             (LPBITMAPINFO)lpD, DIB_RGB_COLORS);

    END_PROCESSING("RANSAC-circle");
}
```

6. Practical work

Using the Diblook framework implement the presented algorithm for fitting a circle to a set of given points. Use the image file *circle.bmp* for loading the input points for the RANSAC circle algorithm. The image has an 8 bit/pixel depth and the points are drawn with black color (intensity value 0). Draw the obtained result over the original image.
The parameters you may use for this lab are: $w = 0.5$, $p = 0.99$, $T=w*n$ (n – total number of points); for t you may try different values like 10, 12, 15 pixels (all these parameters are explained in laboratory 1). Observe how the estimated circle varies with respect to different values of t.

7. References

[1] Alexander Hornberg: *Handbook of Machine Vision*, 2006
[2] Robert C. Bolles, Martin A. Fischler: A RANSAC-Based Approach to Model Fitting and Its Application to Finding Cylinders in Range Data, 1981
[3] Richard Hartley, Andrew Zisserman: *Multiple View Geometry in Computer Vision*, 2003