Pattern Recognition in RICH Counters Using the Possibilistic C-Spherical Shell Algorithm

Anna Maria Massone[1,2], **Léonard Studer**[2] & **Francesco Masulli**[1,3]

(1) INFM - Istituto Nazionale per la Fisica della Materia
via Dodecaneso 33, 16146 Genova, Italy
(2) IPHE - Institut de Physique des Hautes Énergies
Université de Lausanne, BSP, 1015 Lausanne, Switzerland
(3) DISI - Dipartimento di Informatica e Scienze dell'Informazione
Università di Genova, via Dodecaneso 35, 16146 Genova, Italy

e-mail: massone@ge.infm.it, Leonard.Studer@iphe.unil.ch,
masulli@disi.unige.it

Abstract

The pattern recognition problem in RICH counters concerns the identification of an unknown number of imperfect roughly-circular rings made of a low number of discrete points in presence of background. In this paper we present some preliminary results obtained using the Possibilistic C-Spherical Shell algorithm. In particular, we show that the algorithm is very tolerant and robust to noise (outliers rate) level. Moreover, for complex images full of rings, we introduce an iterative scheme that greatly improves performances. Besides that, the rings are not requested to be complete, only arcs are enough to recognize the underlying rings by the algorithm.

1 Introduction

In 2005, the Large Hadron Collider (LHC) at CERN Geneva will be commissioned. Proton against proton will collide at an energy of 14 TeV. Four colliding points will be instrumented. One of the four experiments, LHCb (see `http//lhcb.cern.ch/`), is dedicated to the study of CP violation in the B meson system. CP violation is an ingredient for the explanation of the matter dominance in our Universe. LHCb will use various detectors; among them two RICH counters (Ring Imaging CHerenkov) which will identify the type of stable charged particles (π, K, p, μ and e) produced in the decay of B mesons. For detailed information on RICH, see `http://lhcb.cern.ch/rich/`.

If a charged particle goes through a dielectric material at a speed greater then the speed of light in this material, photons are emitted at a characteristic angle θ from the charged particle flypath. The Cherenkov relation is given by $\cos\theta = 1/\beta n$ where β is the ratio of the speed of the particle to the speed of light in vacuum and n is the index of refraction of the medium. By a clever arrangement of focusing mirrors, it is possible to collect and detect the Cherenkov light emitted by the charged particle on a surface where this light forms circular rings. The ring diameter is a function of θ. Hence measuring the ring diameter allows to measure β and, assuming that another LHCb detector measures the momentum of the charged particle, its type can be derived.

Every 25 ns, in a typical expected pp collision at LHCb, a few tens of charged particles will go through the RICH counters, each one giving a ring whose radius is to be determined. The way the 2 LHCb RICH counters are built implies that some rings will be splitted in two halves on the recorded image; it may even happen that one of the two halves wouldn't hit the sensitive area of photodetectors and hence wouldn't be recorded. Also, the focusing mirrors will distort the Cherenkov cone whose axis will not coincide with the optical axis of the mirrors; this will distort the expected perfectly circular ring in elliptical-like shape. As the Cherenkov process is intrinsically stochastic, the number of recorded photon is a random variable for one given Cherenkov ring (an average about 25 photons per ring). Moreover, charged particles flying backwards will also produce Cherenkov photons. But as these photons are not focused by the optical mirrors (they fly backward), they will appear as blobs (galaxy-like) in the image to be processed. Other sources of noise will set isolated points on the image.

The pattern recognition problem is then to identify an unknown number of imperfect roughly-circular rings made of a low number of discrete points in presence of background points. In this paper, we also assume that the center of the rings are unknown. This last hypothesis is partially realistic as some of these centers could be provided by another LHCb detector with finite efficiency.

Due to the constraints of the problem, distorted circles, partial information, background, randomness and a huge number of image to process, we turn to soft computing algorithms for their robustness, their tolerance to uncertain configurations and their potentially great processing speed.

Our paper will describe our work in progress to tackle this pattern recognition problem. In a first step we have derived an academic model of the underlying problem by testing our algorithm on artificial images made of 3 circles (500 points each) and a variable level of background (10 % to 100 %, which is much higher than expected). In a second step, we will use LHCb simulated event to build realistic images. As this kind of simulation is perfectly mastered in High Energy Physics, we are confident that the simulated images are very similar to the real images that will be recorded.

In the next section we describe in detail the algorithm. Next, we give results for the "3 circles" problem and for the more realistic case of "10 circles". At this stage, it will been shown that adding an iteration loop greatly improved performance which is presented in the fourth section. Finally we end up with our conclusions.

2 The Clustering Algorithm

The initialization step for the algorithm is performed by partitioning the whole set of points by the Fuzzy C-Means algorithm [1]. Any Fuzzy C-Means cluster is potentially a set of points associated to a ring. The center of gravity of the cluster is the initial center of the ring. Projecting the points in the cluster on the X axis gives an histogram where the algorithm searches for the channel with the highest content and the whole set of the adjacent channels whose contents are definitely above the background level. The FWHM (Full-Width Half-Maximum) of this bump is the initial diameter of the ring to be recognized. Starting from these initial value for the center positions and radiuses, the Possibilistic C-Spherical Shell algorithm [4, 2] searches for prototypes consisting of two parameters $(\mathbf{c_j}, r_j)$, where $\mathbf{c_j}$ is the center and r_j the radius of the j-th circle. Given a generic point $\mathbf{x_k}$, belonging to a set of N samples, a distance measure from the prototype can be defined as:

$$d_{jk}^2 = d^2(\mathbf{x_k}; \mathbf{c_j}, r_j) = (\|\mathbf{x_k} - \mathbf{c_j}\| - r_j)^2 \qquad (1)$$

In order to obtain the final mapping $\mathbf{x_k} \rightarrow (\mathbf{c_j}, r_j)$, the objective function iteratively minimized is [3]:

$$J(L, \mathbf{U}) = \sum_{j=1}^{C}\sum_{k=1}^{N} u_{jk} d_{jk}^2 + \sum_{j=1}^{C} \eta_j \sum_{k=1}^{N} u_{jk}(\ln u_{jk} - 1) \qquad (2)$$

where L is the set of C prototypes and $\mathbf{U} = [u_{jk}]$ is the membership values matrix. Using the possibilistic approach, the elements u_{jk} of $\mathbf{U}$ will fulfill the following relaxed conditions:

$$\begin{aligned} & u_{jk} \in [0,1] && \forall j,k \\ & 0 < \textstyle\sum_{k=1}^{N} u_{jk} < N && \forall j \\ & \max_j u_{jk} > 0 && \forall k \end{aligned} \qquad (3)$$

It can be shown [4] that rewriting the distance (1) as

$$d_{jk}^2 = \mathbf{p_j^T M_k p_j} + \mathbf{v_k^T p_j} + b_k \qquad (4)$$

where

$$\begin{aligned} & b_k = (\mathbf{x_k^T x_k})^2 && \mathbf{v_k} = 2(\mathbf{x_k^T x_k})\mathbf{y_k} \\ & \mathbf{y_k} = \begin{bmatrix} \mathbf{x_k} \\ 1 \end{bmatrix} && \mathbf{M_k} = \mathbf{y_k y_k^T} \\ & \mathbf{p_j} = \begin{bmatrix} -2\mathbf{c_j} \\ \mathbf{c_j^T c_j} - r_j^2 \end{bmatrix} \end{aligned} \qquad (5)$$

the vectors $\mathbf{p_j}$ minimizing the objective function (2) are given by:

$$\mathbf{p_j} = -\frac{1}{2}(\mathbf{H_j})^{-1}\omega_\mathbf{j} \qquad (6)$$

where

$$\mathbf{H_j} = \textstyle\sum_{k=1}^{N} u_{jk}\mathbf{M_k} \quad \omega_\mathbf{j} = \sum_{k=1}^{N} u_{jk}\mathbf{v_k} \qquad (7)$$

and the update equations for u_{jk} are:

$$u_{jk} = \exp\left(-\frac{d_{jk}^2}{\eta_j}\right) \qquad (8)$$

where the scale parameter η_j corresponds to the zone of influence of a cluster.

For the η_j parameter, we did not use the original definition given by authors [2], but we introduced an ad hoc definition for the case of spherical shell detection.

Interpreting η_j as the distance at which the membership value of a point in a cluster becomes 0.5, we introduced the parameter p representing this distance as function of a radius percentage $(r_j \pm p r_j)$.

At this distance:

$$\frac{1}{2} = \exp\left(-\frac{p^2 r_j^2}{\eta_j}\right) \qquad (9)$$

Table 1: Results of several trials in presence of different values of noise.

noise (%)	Trials	Successes	Mistakes
0%	20	20	0
10%	20	20	0
20%	20	20	0
30%	20	19	1
40%	20	20	0
50%	20	20	0
60%	20	17	3
70%	20	20	0
80%	20	19	1
90%	20	19	1
100%	20	0	20

and so

$$\eta_j = \frac{p^2}{\ln 2} r_j \tag{10}$$

We set the value of p at 30%.

3 Experimental Dataset and Results

First we want to estimate the intrinsic robustness of the algorithm in function of noise level (outliers rate). In a basic test of the algorithm, we generated a database consisting of 3 circles (each circle containing 500 points), whose centers and radiuses have been set to $\{\mathbf{c_j}, r_j\} = \{(2,3),1\}; \{(4,5),0.5\}; \{(2.5,6),1\}$.

We added to database points the noise at various levels (0% up to 100%) and we tested the algorithm 20 times for each noise value. The initial number of circles has been set to 3. The algorithm appears to be very strong to the noise level as we can see in Tab. 1.

For the cases of 10% and 90% of noise, in Fig. 1 the initial partitioning performed by the Fuzzy C-Means is shown. The initial position of centers (markers 1, 2, 3) is set at random and the Fuzzy C-Means iterates a few times to estimate better center positions, this is visible as big jumps on the figure. Next, the final small jumps are the centers estimated by the Possibilistic C-Spherical Shell step. The recognized circles are perfectly superimposed on database points. For this particular cases, we obtained the values for centers and radiuses shown in Tab. 2. We mention that no direct regression has been carried on the points to measure the radiuses; this estimation of radiuses is then a pure side product of the pattern recognition step itself.

Table 2: Centers and Radiuses found when noise=10% and noise=90%

10%		90%	
$\mathbf{c_j}$	r_j	$\mathbf{c_j}$	r_j
$(2.000, 2.999)$	1.000	$(2.002, 2.994)$	1.013
$(3.989, 5.005)$	0.505	$(3.983, 5.008)$	0.510
$(2.501, 6.000)$	1.001	$(2.512, 5.990)$	1.010

4 Additional Iteration Loop

We present here a more realistic database consisting of 10 circles whose radiuses may have values between 0.5 and 1. Each circle is made of 500 points and we considered only the case with a level of 10% of noise. Initial number of circles has been set to 10. As we can see in Fig. 2(a), the algorithm finds exactly 6 circles (the ones marked from 1 to 6). Circle 7 is found with a small error (expected: $\mathbf{c} = (2.80, 5.28)$ $r = 0.56$; found: $\mathbf{c} = (2.89, 5.28)$ $r = 0.61$). The remaining 3 circles are completely wrong.

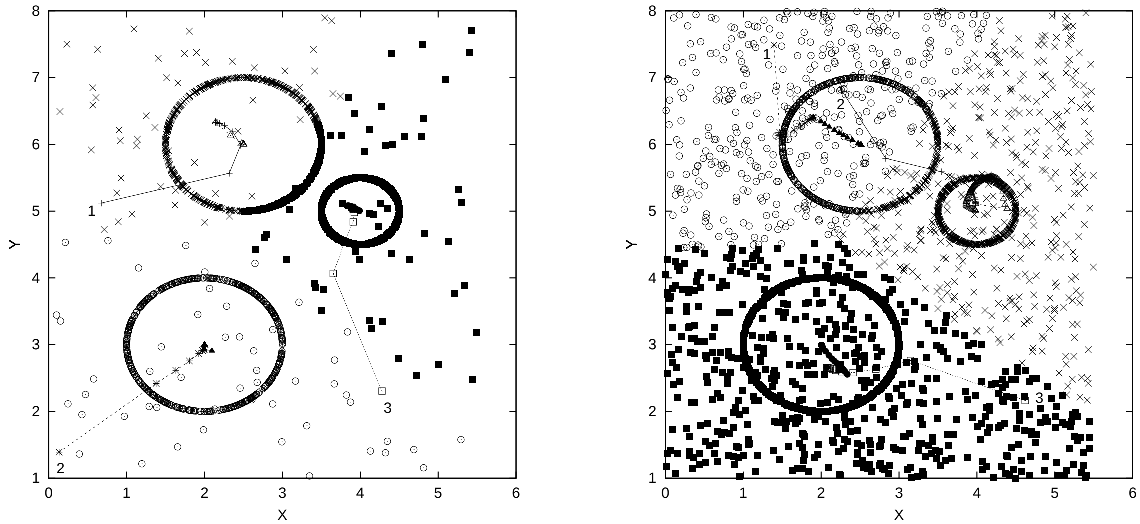
But at this stage, it is easy to distinguish between "good" and "bad" circles because noisy circles are generally less densely populated than the good ones and most points belonging to correct circles have very high membership values (~ 0.99).

We introduced two α-cuts, the first one in order to remove the noise ($\alpha_1 = 0.1$) and the second one in order to remove the points already correctly classified ($\alpha_2 = 0.96$).

The new "cleaned" database is now consisting of 3 circles with holes and arcs that are lost fragments of circles already recognized and removed.

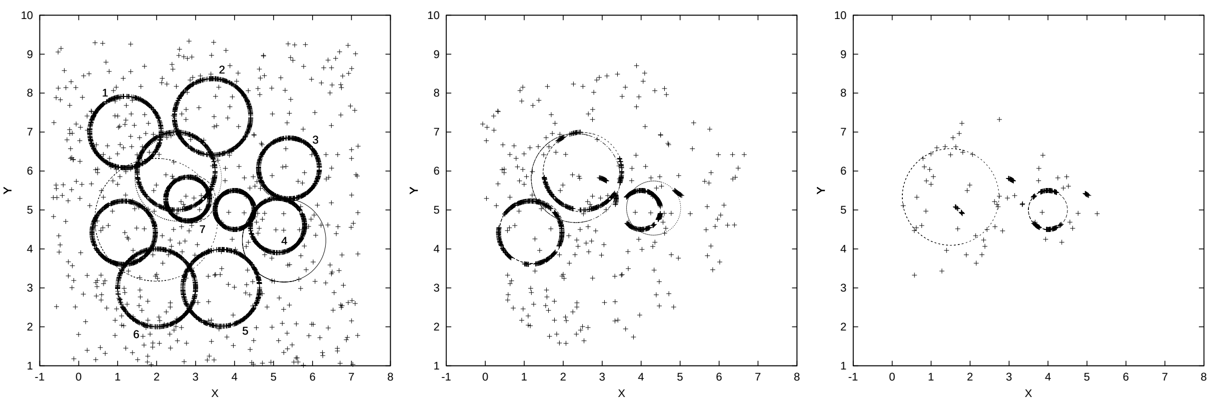
We reiterated the algorithm searching for 4 circles and at this second step we found 2 other circles (Fig. 2(b)). An additional iteration step is necessary in order to find the last circle (Fig. 2(c)). Removing the patterns already classified and iterating the algorithm three times, we have recognized the whole set of circles.

Figure 1: Solutions found when (a) noise=10% (b) noise=90%.



(a) (b)

Figure 2: Iteration steps with a database consisting of 10 circles



(a) (b) (c)

5 Conclusions

We have presented results using the Possibilistic C-Spherical Shell algorithm to recognize rings on images. These rings are roughly circular, built with discrete points and noise randomly scattered in the image. The algorithm presented has been shown to be very tolerant and very robust to noise level.

The fuzziness included in the algorithm adds tolerance to the imperfect circular rings. One tunable parameter η controls the level of tolerance. We have introduced a very simple scheme of iteration that perfectly recognizes complex images full of circles.

It is worth noting that the rings are not requested to be complete (see Fig. 2(b) e (c)); only arcs are enough to recognize the underlying rings. This automatically solves the realistic problem of half circles mentioned in the introduction.

We have also noticed that overestimating the initial number of circles doesn't degrade the efficiency performance of the algorithm. As a point is not excluded to be in more than one circle (possibilistic membership), 2 circles can coexist on the same set of points. In other words, extra circles collapse to expected circles.

References

[1] J.C. Bezdek. *Pattern Recognition with Fuzzy Objective Function Algorithms.* Plenum Press, New York, 1981.

[2] R. Krishnapuram and J.M. Keller. A possibilistic approach to clustering. *IEEE Transactions on Fuzzy Systems*, 1(2):98–110, 1993.

[3] R. Krishnapuram and J.M. Keller. The possibilistic c-means algorithm: Insights and recommendations. *IEEE Transactions on Fuzzy Systems*, 4(3):385–396, 1996.

[4] O. Nasraoui R. Krishnapuram and H. Figui. The fuzzy c spherical shell algorithm: A new approach. *IEEE Transactions on Neural Networks*, 3(5):663–671, 1992.