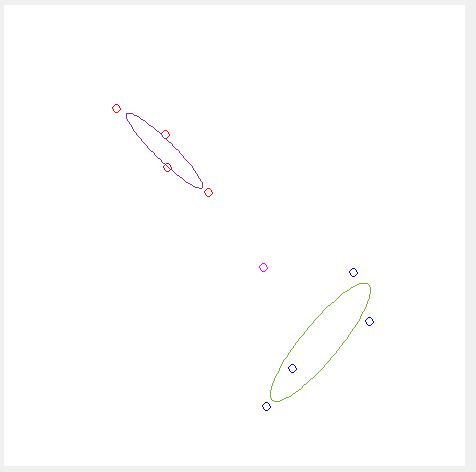
# COMPUTER VISION

**EXERCISE 6a: Object recognition with mVision**

Concepts: Bayesian Classifier

1. **Running the GUI:** Launch the **recognition** GUI of mVision, and follow the instructions on the bottom part of it to insert objects (points) in a similar way to how they are shown in the following figure (try to insert them in similar positions). The red points represent two features of objects of one class, the red ones features for other class, and the magenta point is the object to classify:

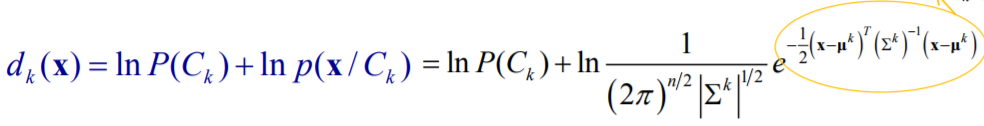


1. **Recognition:** Push the **Recognize** button and interpret the results obtained on the right part of the GUI (decision functions, covariance matrices, etc.).
2. **Same probability and covariance:** Now force the classes to have the same probability. With that the GUI also forces the classes to have the same covariance matrices. This emulates situations where both classes have the same dispersion. Discuss the new results.
3. **Isotropic covariance:** Force them to have isotropic covariance. Discuss the obtained results.
4. **Distances:** Finally, comment which distances have been used in each section (2 to 4) to compute the decision functions: Mahalanobis or Euclidean?

**EXERCISE 6b: Implementing a classifier**

Concepts: Bayesian Classifier

1. **Centroid and covariance computation:** Develop a code that computes the statistical parameters (centroid and covariance matrix) of the Hu moments of a set of 15 images of three different bottle classes. Use the function calculating the Hu moments from previous exercises.
2. **Visualization:** Show graphically the centroid and covariance of each class (only the two first Hu moments). Use the function **error\_ellipse** to represent the covariance matrix. Which information could we get from that ellipses? *Note: you can multiply the covariance matrices by a scalar for improving their visibility.*
3. **Bayesian classifier:** Design a classifier that takes as input the image of a bottle and assigns it to its corresponding class. For doing so:
   1. Implement a function called **decision\_function** that computes the expression **dk(x)** shown below. The input of this function should be: the 7 Hu moments of the initial image, and the centroid **k** and covariance **k** of the class Ci.



* 1. Write a function called **belonging\_class** that calculates and displays the values of the decision function for the three different classes, and returns the belonging class of the input image. Use the **max** function to determine the class.

1. **Classifying**: Extend the previous script with a loop reading each of the 5 unused images of the three bottle classes, and:
   1. represents in the figure the two first Hu moments of each image,
   2. shows the assigned class. Use the function **text** to display a text in the (x,y) coordinates (show the image). *Note: don’t forget the pause and delete functions.*
   3. Comment the obtained classifications.
2. **Optional exercise!**: Retrieve the decision boundaries between the pairs of classes and show them graphically over the figure using the given function **plotConic.m**. Employ only the two first Hu moments (it can not be represented using the 7 moments). The decision boundaries are obtained using the following expression:

where *dij* is the boundary between classes *i* and *j,* and is the decision function of the class *i* assuming that we only employ 2 Hu moments ( and ).

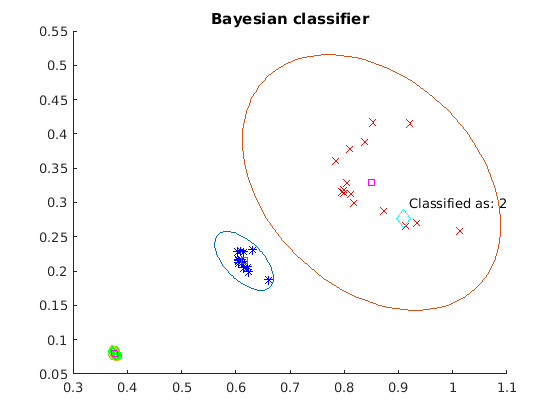
**Notes:**

The prior probability is the same for all the classes (p(Ci )= 1/3, i=1,2,3).

**Commands:**

|  |  |
| --- | --- |
| **error\_ellipse(C,m)** | Draws an ellipse representing the covariance matrix **C** centered at the point **m**. |
| **text(x,y,’text’)** | Shows in the coordinates **(x,y)** the **text** between colons. |
| **plotConic(K,color,xBounds,yBounds)** | Draws the conic curve given by the matrix **K**. |

**Results**

****