Github link of project (contain training dataset that didn't fit the file size limit on ISIS):
https://github.com/noranMoh/RoboCup2020

Abstract

As it is the most important object in a soccer game, being able to detect the ball is vital for a soccer playing NAO robot. Detecting the ball is always considered a challenge in a competition such as RoboCup. Years ago, the RoboCup junior category even used an infrared ball to make it easier to detect without even using a camera. However, as engineers always aspire to make robots as capable as humans -or even better- normal balls started to be used in competitions increasing the challenge of detecting it. A few years ago, soccer playing robots used to use a uni-colored ball during their games. A bright orange ball was used in the RoboCup competitions. Detecting this kind of balls was dependant on its unique color. Nowadays, the competition is held with the normal black and white ball making it not possible to detect the ball by its unique color and a classifier would be needed for the robot to be able to recognise this type of balls. In this project, both the orange and the black and white balls will be considered and the different methods of detecting them will be explored.

Orange Ball detection

My first attempt to detect the ball was to try to detect a uni color ball similar to the ball used in RoboCup competitions before. Detecting a uni color orange ball is easier than detecting the normal black and white ball because it can be detected using its unique color. I started by finding the values of the orange color in the HSV color space. Using color.py file I was able found out these values by showing an orange ball to my webcam choosing which part I would wnt to know the HSV values of. I then set the upper and lower bounds of the orange color according to the outputted values. Since I did not have access to the NAO humanoid robot I used the laptop's webcam to detect the ball using openCV. First, the image is pre-processed by reducing the frame size to be able to process the frame faster, blurring it to reduce high frequency noise and finally converting the frame to HSV color space. Then the parts of the frame that belongs to the preset range of the color orange are detected using cv2.inrange. Some morphological operations are then applied such as erosions and dilations to remove any small blobs that may be left on the mask. We then check to ensure at least one contour was found in the mask . Provided that at least one contour was found, we find the largest contour and compute the minimum enclosing circle of the blob, and then compute the center (x, y)-coordinates. Knowing the coordinates of the ball and the position of the camera that could be deduced from the angles of the head's joints, the robot would be able to know in which direction it should move to get closer to the ball. Using the coordinates a circle is drawn around the ball tracking it to be visible in the video. The video orange_ball_detection shows the tracking of the orange ball.

Black and white ball detection

As people are attempting to make robots behave more and more as humans, the normal black and white ball was introduced to the RoboCup competition. This made detecting the ball more challenging as we can no more search for a specific color in our frame. This is why using other features should be considered as the color feature is now unreliable. One method to do so is to train and use a classifier. I chose to train the openCV cascade classifier. This calssifier is a Haar

feature-based cascade classifier. Haar feature based classifier uses 3x3 matrices (kernel) that moves over the image and perform matrix multiplication with every 3x3 part of the image, emphasizing some features and smoothing others. Some of the haar classifier with their numerical representations are shown in figure 1. The first two detect edges, the third is the line feature and the fourth is mostly used to detect an inclined line.

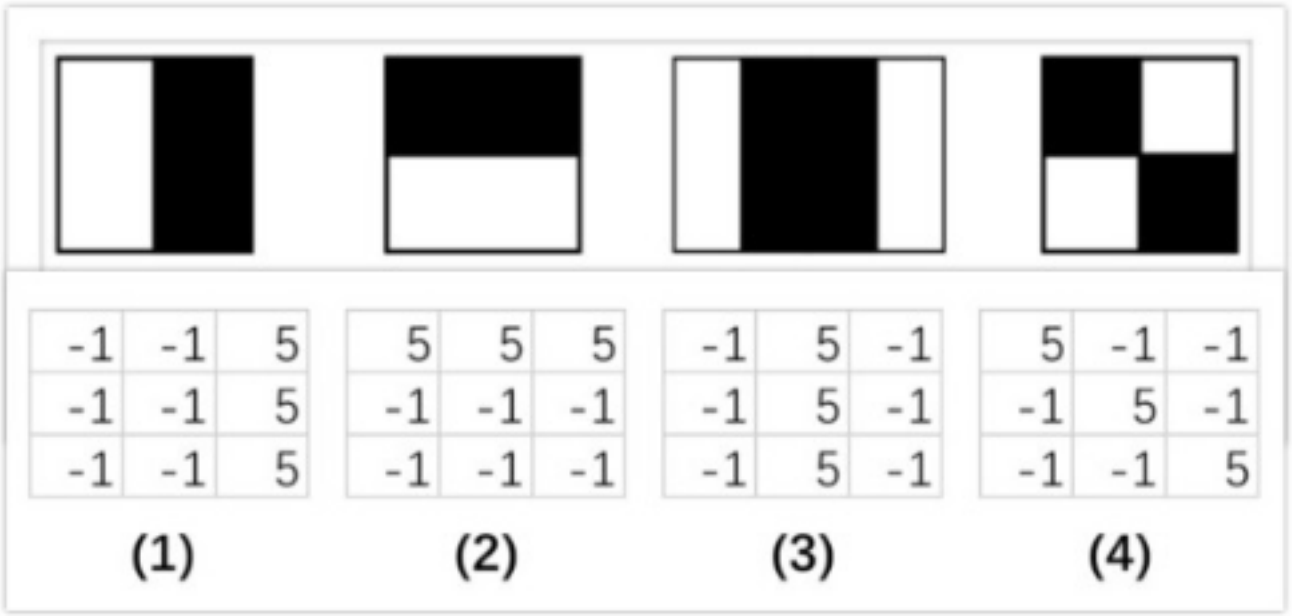

Abbildung 1: Common Haar features[4]

This calssifier has several advantages: Since the feature does not need to be trained, a classifier could be created with smaller data set all we have to do is to train the weightings of each feature to know which features should be used more. Moreover, this classifier has a higher execution speed as Haar-based classifiers often involve less calculations. The openCV usese a cascade Haar classifier. The idea of cascade classifiers is to concatenate weak classifiers to build a strong classifier. Instead of applying all 6000 features on a window, the features are grouped into different stages of classifiers and applied one-by-one. If a window fails the first stage, it is discarded. The remaining features are not considered. If it passes, the second stage of features are applied to it and the process is continued. The window which passes all stages has the object that is being detected. The classifier need both positive and negative data sets, where positive samples are images with the object present and negative ones are image samples without the object. The set of negative samples must be prepared manually, whereas set of positive samples is created using the opencv_createsamples application. First I had to have both positive and negative data sets. Since I did not have access to a NAO robot and pictures taken by its cameras, I downloaded pictures of photos taken by the NAO robot top cameras[1]. I used the openCV cascade training setting the number of stages to 20, the minimum hit rate to 0.999, maximum alarm false rate to 0.5 and setting the nonsymmetrical parameter as the ball is not symmetrical and I began the training process. I then used the trained classifier in detecting the ball. Since again I do not have access to a NAO's camera, I downloaded images taken by NAO robots[1] or random photos taken in NAO soccer games[3] to test my classifier. The photos and the results after detection are found in the folder test. The video black_and_white_ball_tracking shows detection of ball using webcam.

Resources

1. http://www.dis.uniroma1.it/~labrococo/
2. http://spqr.diag.uniroma1.it
3. https://commons.wikimedia.org/wiki/File:SPL_Team_B-Human,_RoboCup_2016.jpg
4. https://medium.com/analytics-vidhya/haar-cascade-face-identification-aa4b8bc79478