Monitoring system for Robotic Soccer

Objectives: Build a Computer Vision system that is able to monitor a robotic soccer game. It should be able to track the most relevant moments in robotic soccer games and provide their statistics. For that end, the system must use suitable Computer Vision algorithms for fundamental tasks such as distinguishing the teams and the referee, tracking the ball and the scored goals, etc.

Deliverable 1 (26/02/2020):

1. Acquire images using the Raspberry camera or webcam connected to your computer.
2. Explore saving videos using compression algorithms (ex. H.264, MJPEG, etc.).
3. Apply color calibration (intensity normalization, white balance, etc).
4. Include a watermark (it can be a string identifying the group or a chosen picture).
5. Video player able to work with the camera in real-time and read video files.

Deliverable 2 (11/03/2020):

1. Transform the acquired images to other color spaces, namely YUV and HSV.
2. Calculate and display the histograms in real-time of the acquired and transformed images.
3. Convert the acquired images to grayscale and apply histogram equalization.
4. Apply gaussian and blur filters to the acquired images, exploring different filter kernels.

Deliverable 3 (25/03/2020):

1. Interact with the OpenCV windows, namely using the mouse (you have to implement the method `cv.setMouseCallback()`) and trackbars (see for example the methods `cv.getTrackbarPos(), cv.createTrackbar()`).
2. Segment the most important colors of the CAMBADA soccer field based on color threshold. This threshold is controlled by the developed trackbars in the previous exercise.
3. Perform object detection on grayscale images, resulting from the previous segmentation, morphological operators and low level image features (ball, goals, soccer lines, robots, referee / people, etc).
4. **Extra mile:** Explore automatic segmentation algorithms (e.g. watershed, region growing, etc).

Deliverable 4 (15/04/2020):

1. Find image gradients (Sobel, Scharr and Laplacian derivatives).
2. Apply the Canny edge detection algorithm, exploring its parameters. Explore how to manipulate the corresponding contours.
3. Perform object detection using contour detection.
4. Perform line detection and ball detection using the Hough transform (ex. this should allow the detection of balls with different colors and balls in the air).
5. **Extra mile:** Explore other object detection algorithms (e.g. Machine Learning).

Deliverable 5 (06/05/2020):

1. Perform intrinsic and extrinsic camera parameter calibration using a chess board.

Deliverable 6 (20/05/2020):

1. Explore the Lucas-Kanade optical flow algorithm to perform object tracking. Using the developed object detection algorithms, the software must be able to distinguish which objects are being tracked (multiple balls and robots; assigning an unique ID, etc).
2. Estimate the travelled distance of the ball and teams during a game.

Deliverable 7 (June): Demo in a soccer game between CAMBADA and a human team. Build an application that is able to:

1. Detect and track the robots of CAMBADA, the humans, the referee and the ball.
2. Estimate the travelled distance of the ball, referee and teams during the game.
3. Detect successful passes between the players, goal attempts, goalkeeper saves and goals.
4. Output the game statistics at the end of the game.