Billiard table segmentation, balls location, segmentation and tracking

Table segmentation (Artico Giovanni)

The following assumptions were made for the segmentation of the table

- the table is a uniform color and is a prevalent part of the image
- the table is centered (or close to being centered) in the image

It is going to be apparent the reason for these when the full algorithm for the segmentation of the table is explained. The first thing that was to be found was a rough segmentation of the table, as finding the lines with only the Canny-transform of the full image would be problematic as the strongest lines in the image are not only the table fabric but also the table itself and various other things in the image. The initial approach was to use Otsu's thresholding, as the table is a uniform color, therefore it is sensible to assume it would be thresholded the same way throughout. This proved unfruitful as the segmentation was too rough. The second approach was to use kmeans clustering, in particular exploiting the adaptability of the algorithm in order to use spatial features too to improve the segmentation. Out of the k-clusters obtained the most centered one was picked. THe segmentation obtained was fairly precise but contained multiple disconnected components, however we can consider the table the biggest and pick it, in particular before doing this operation erosion is applied in order to remove weakly connected objects. the k (empirically) chosen was 3, as 2 lead to undersegmentation of the image and 4 to oversegmentation. The resulting segmentation is precise in most cases, but given the nature of kmeans on some images it fails, as it clusters some unrelated parts of the table with the fabric, due, for example, to lighting. For this reason further processing was required, given we have obtained a rough mask of the table we now obtain the mean color of the image in the pixels where the mask is non-zero. After this we use a color thresholding by distance from the mean and the same way we did before to obtain the largest connectedcomponent. In particular thresholding on the hue is the most robust, in particular a low distance, 5, was chosen, but similar distances give similar results.

after having obtained this result dilation is applied in order to remove some noise from the mask, for example balls or other objects. Afterwards the hough transform is used to find the lines, with a threshold in order to remove similar lines. The vertices of the table are found using the equations from https://en.wikipedia.org/wiki/Line%E2%80%9

side recognition (Artico Giovanni)

the points are ordered from the upper-left-most point clockwise, but this doesn't give any information on how the sides are ordered. It is unfortunately impossible (or out of my mathematical capabilities) to find any meaningful geometrical relation between the sides given the perspective deformations in some cases. The only feature we can go off is the holes in the middle of the longer sides. The first operation was to obtain the rotated rectangles containing the sides of the pool table, with a small width as to avoid keeping in the images unneeded information.

Multiple approaches were tested to distinguish the two types of sides. The first was to use features (such as harris corners or SIFT), as intuitively they should appear on the holes and not on uniform sides as they should be classified as edges. This didn't work, as the features and corners detected most of the time were spread throughout the considered sides most of the times.

The next approach was to try template matching between different segments of the image, as the center contains a hole which should make the match significantly worse than between the left and right segment. This does not work for two reasons:

- because of the perspective the hole can appear in a part of the segment of the image that is not the center one
- because of the illumination and small imprecisions on the table segmentation the matching would fail even on the segments which should result similar

Ultimately the way to recognize the sides was the following:

- apply the Canny tranform on the initial image
- extract the rectangles from the transformed image

- rotate the rectangles so that they are horizontal
- apply sobel on the x axis
- sum over the contribution on the middle two quarters of the image (normalized over the number of pixels contained)
- determine the longer sides by the highest contribution

This works because of the nature of the edges on the sides: on shorter sides there are only parallel lines, instead on longer there are strong lines orthogonal given by the holes. Only the middle two quarters are considers as to avoid counting the corner holes as edges.