CS4243 Cheatsheet AY22/23 Sem2

Content

Intensity:

I = (R+G+B)/3

Scaling intensity -> increase contrast and vice versa

Whitening:

$$\mu = \frac{\sum_{ij} p_{ij}}{IJ}, \quad \sigma^2 = \frac{\sum_{ij}(p_{ij}-\mu)^2}{IJ}$$

Then $x_{ij} = \frac{p_{ij}-\mu}{\sigma}$

RGB to HSV:

$$H \in [0{,}360], S \in [0{,}1], V \in [0{,}255]$$

Histogram Stretching:

$$H = \begin{cases} \frac{G-B}{V-\min\{R,G,B\}} \cdot 60°, & \text{if } V = R \text{ and } G \geq B; \\ \left(\frac{B-R}{V-\min\{R,G,B\}} + 2\right) \cdot 60°, & \text{if } G = V; \\ \left(\frac{R-G}{V-\min\{R,G,B\}} + 4\right) \cdot 60°, & \text{if } B = V; \\ \left(\frac{R-B}{V-\min\{R,G,B\}} + 5\right) \cdot 60°, & \text{if } V = R \text{ and } G < B \end{cases} \qquad H \in [0°, 360°[$$

$$S = \frac{V - \min\{R,G,B\}}{V} \quad S \in [0,1]$$

$$V = \max\{R,G,B\} \quad V \in [0,255]$$

$$x_{ij} = \frac{p_{ij} - min(p)}{max(p) - min(p)} \times [255]$$

Histogram Equalization:

P_{ij}	count	%	Cum%	X_ij = Cum% x max(p)

Filters:

Sharpening

- Does nothing for flat areas
- Amplify noise

Shift Up Kernel, Shift Left Kernel

$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1.5 \end{bmatrix}, \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}$$

Smoothing/Blur Kernel

$$\frac{1}{n}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

Horizontal and Vertical Sobel

$$S_x = \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix}, S_y = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$$

- Horizontal -> vertical lines
- Combines blur + derivative filter
- Increase sigma -> detect larger edges and vice versa

X-Cor vs Convolution:

Convolution flips the kernel in both axis, then does X-Cor
Both are equivalent if kernel is symmetric

K-Means Clustering:

Sensitive to outliers, initial clusters
Need some validation for setting k
Bias towards spherical clusters

Mean-Shift Clustering:

Parameter: window size a.k.a bandwidth
Changing gaussian mean does not affect it but variance does
Robust to outliers & no assumption on #clusters, cluster shape vs K-Means

SLIC Superpixels:

Define $n_{tp}, n_{sp}, S = [n_{tp}/n_{sp}]^{1/2}$
Initialize superpixels as grid of (H/S, W/S)

Canny Edge Detector:

1. Filter image with DoG
2. Find magnitude and orientation
3. Perform NMS
4. Edge linking via hysteresis thresholding

Scaling I eg I' = aI + b does not affect the output if relative threshold is applied during hysteresis linking
Non linear transforms like histogram equalization will cause output to be different!

Hough Transform (Normal Form):

```
1.   Quantize Param Space ($\theta$,$\rho$)
2.   Create Accumulator Array A($\theta$,$\rho$)
3.   Initialize all elements to 0
4.   For each (xi, yi)
         a.   For each $\theta$
                Solve $\rho = x_i cos\theta + y_i\, sin\,\theta$
                $A(\theta,\rho) = A(\theta,\rho) + 1$
5.   Threshold, find local max.
```

Keypoints:

Invariance – image transformed and detection (score) same
Equivariance – detection (location) undergoes a similar transformation

Good Keypoints:

Repeatable – same despite transformations
Distinct – captures different information
Local – Covers only a small area of image

Harris Corners:

1. Precompute gradients
2. Compute H matrix for window around each pixel

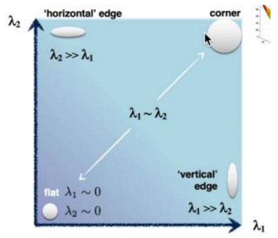
$$H = \sum_{(x,y)\in W} w_{x,y} \begin{bmatrix} Ix^2 & I_x I_y \\ I_x I_y & Iy^2 \end{bmatrix}$$

3. Compute score $R = det(H) - k(tr(H))^2$
4. Threshold R > (threshold)
5. NMS

Interpretation of eigenvalues of H:

λ_{max} – direction of fastest change
λ_{min} – direction of slowest change

Corners have eigenvalues both sufficiently large!



Invariance:
- Translation
- Rotation
- (NOT) Photometric (scaling, contrast)
- (YES) Addition (brightness)

Good Descriptors:

Invariant/equivariant – same after transform
Discriminative – can differentiate points.

What to use as Descriptors?

Intensity, derivatives (gradient information), color histogram, spatial histogram

SIFT Descriptors

Robust to changes in viewpoint, illumination, scale & rotation. + it's fast

Accuracy Metrics

TP	Correct match
FP	Wrong match
FN	Feature pairs not matched
TN	Not part of any feature pair
Precision TP/(TP+FP)	Accuracy of detected pairs
Recall TP/(TP+FN)	Ability to find all actual pairs of features
Specificity TN/(TN+FP)	Ability to disregard features not part of any pair

Applying Homography

heterogeneous coordinates　　homogeneous coordinates

$$p = \begin{bmatrix} x \\ y \end{bmatrix} \Rightarrow P = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Add a 1 here

What is the size of the homography matrix?　[3x1] = [3x3][3x1]

$$P' = H \cdot P$$

$$P' = \begin{bmatrix} x' \\ y' \\ w' \end{bmatrix} \Rightarrow p' = \begin{bmatrix} x'/w' \\ y'/w' \end{bmatrix}$$

Solving for H via DLT

1. For each correspondence, create 2x9 matrix $\mathbf{A}_i$
2. Concatenate into single 2n x 9 matrix $\mathbf{A}$
3. Compute SVD of $\mathbf{A} = \mathbf{U\Sigma V}^\top$
4. Store singular vector of smallest singular value $\boldsymbol{h} = \boldsymbol{v}_{\hat{i}}$
5. Reshape to get $\mathbf{H}$

$$\mathbf{A}_i = \begin{bmatrix} -x & -y & -1 & 0 & 0 & 0 & xx' & yx' & x' \\ 0 & 0 & 0 & -x & -y & -1 & xy' & yy' & y' \end{bmatrix}$$

Drawbacks:
DLT requires transformation btw 2 images to be LINEAR
Sensitive to outliers -> Use RANSAC
Sensitive to scaling

RANSAC:

1. Randomly get (min 4) correspondences
2. Compute H using DLT
3. Count inliers
4. Keep H if largest number of inliers

Recompute for best H using all inliers

Can estimate the number of iterations required:

$$N = log(1-p) \,/\, log(1-(1-e)\text{^}s))$$

e is probability that a point is outlier
s is number of points drawn per iteration for fitting
p is probability that at least one set of points does not contain any outliers (0.99 or given)

Optical Flow:

Assumes color constancy and small motion.
Solve $I_x u + I_y v + I_t = 0$
Solutions (u,v) lie on straight line – not unique – require additional constraints

Lucas-Kanade – assume constant flow for all pixels, sparse flow field
Horn-Schunk – every pixel has its own (u,v), dense flow field, enforce smoothness, small λ means high smoothness

Translation

$$\mathbf{W}(\boldsymbol{x};\boldsymbol{p}) = \begin{bmatrix} x + p_1 \\ y + p_2 \end{bmatrix} = \underbrace{\begin{bmatrix} 1 & 0 & p_1 \\ 0 & 1 & p_2 \end{bmatrix}}_{\text{transform}} \underbrace{\begin{bmatrix} x \\ y \\ 1 \end{bmatrix}}_{\text{coordinate}}$$

Affine

$$\mathbf{W}(\boldsymbol{x};\boldsymbol{p}) = \begin{bmatrix} p_1 x + p_2 y + p_3 \\ p_4 x + p_5 y + p_6 \end{bmatrix} = \underbrace{\begin{bmatrix} p_1 & p_2 & p_3 \\ p_4 & p_5 & p_6 \end{bmatrix}}_{\text{affine transform}} \underbrace{\begin{bmatrix} x \\ y \\ 1 \end{bmatrix}}_{\text{coordinate}}$$

< Some examples of 2D Transformations

PYP

Repeatedly applying a blur filter to image

All pixels converge to uniform value = average of input image
If image is zero padded, converge to 0 because repeated smoothing introduces 0 values in the borders.

X-Cor without Normalization

Brightest patch in the image will give the strongest response, same for convolution!

Adjusting Canny for Different Res for Same Image

Adjust the sigma used for the Gaussian smoothing applied in Canny (either as part of the DoG filter when finding the gradients or the separate Gaussian smoothing before applying gradient filters). For some fixed sigma used to get the results in (iv) to (vi), we should decrease the sigma for the half and quarter res images to match the results of (iv) (column 4 below). To match the results in (vi), we need to increase the sigma of the full and half-resolution images

Detection “straight” tracks angled at ±30° and ±60° off the horizontal axis. There is an allowed estimation error of ±5° in the tracks and your method should be able to distinguish two tracks which
are a minimum of 3 pixels apart. For a track to be valid, it must be at least 100 pixels long.
Method (1 point): Use standard voting for line detection, with the normal form of parameterization (theta /
rho). The tracks of interest will appear in the 4 theta bins for ±30° and ±60°. To 100 pixel long lines,
threshold the accumulator array to only consider bins with > 100 votes.

Accumulator Array (1 point): the voting space is 2D, with parameters theta and rho. The smallest accumulator
array is 4 x (2D/3+1): 4 bins for theta cover from [-65:-55, -35:-25, 25:35, 55:65] degrees, 2D/3+1 bins for rho
ranging from [-D:3:D], where D is the diagonal length of the image.

Assumptions: Thresholding for bins with 100 votes or more assumes that every voting pixel contributes to the
line. However, votes are likely to come from noise pixels in the background. This is especially the case if we
use coarse bins based on the minimum accumulator array.

Method to segment the pool balls so that the reflections on the balls are not grouped together with the background.
SLIC first; apply k-means clustering on the superpixels.
Rationale: if we replace the reflection pixels with other RGB values that are more similar in colour to the ball, then it won’t be mis-segmented.
Tradeoff: the segmentation may no longer adhere as well to the original image boundaries and contours, e.g. superpixels shapes may not follow ball exactly; blurring would completely
lose the image edge

Approach based on filter banks to isolate all the pool balls. What types of filters would you include and why?
Consider two forms of textures – the sponge background, or the balls themselves. The filter banks should include very small scaled filters over a range of orientations tuned towards the sponge, and very large difference-of-Gaussian filters in approx. same scale as the balls.

Rationale is that both the balls and the background can be treated as a texture.
Assumption: the image can be contrastenhance sufficiently that we can get a reasonable response on the sponge background, as it is already very bright white colour
Descriptor Invariances

• Transformation: $I'(x,y) = I(x+\Delta u, y+\Delta v)$ where $\Delta u > 0$, $\Delta v > 0$
custom, Harris, LoG, SIFT (all are invariant to position offsets)
weighted intensity, weighted gradients, GIST, SIFT (all except GIST)
• Transformation: $I'(x,y)=I(x,y)+\Delta u$, where $\Delta u > 0$
custom, Harris, LoG, SIFT (all are invariant to intensity offset)
weighted intensity, weighted gradients, GIST, SIFT (all except weighted intensity)
• Transformation: A mirrored version i.e. flipped over the X axis.
custom, Harris, LoG, SIFT (all are invariant to flipping)
weighted intensity, weighted gradients, GIST, SIFT (GIST and SIFT are not mirror invariant)
• Transformation: x,y -> y,x
custom, Harris, LoG, SIFT (all are invariant to transpose)
weighted intensity, weighted gradients, GIST, SIFT (GIST and SIFT descriptors are not invariant)
• Transformation: x,y -> 0.5*x, y
custom, Harris, LoG, SIFT (none)
weighted intensity, weighted gradients, GIST, SIFT (none)
• Transformation: x,y -> 0.5*x, 0.5*y
custom, Harris, LoG, SIFT (scaling transformation)
weighted intensity, weighted gradients, GIST, SIFT (only SIFT)
• Transformation: $III\ xx,\ yy$ is a histogram stretched version of to $II\ xx,\ yy$.
custom, Harris, LoG, SIFT (none are invariant)
weighted intensity, weighted gradients, GIST, SIFT (none)
• Transformation: $III\ xx,\ yy$ is 45 degree clock-wise rotated version of $II\ xx,\ yy$.
custom, Harris, LoG, SIFT (all except custom)
weighted intensity, weighted gradients, GIST, SIFT (all except GIST)

Detect individual bottles to find n.
Scanline solution:
It is stated that the background is black, the bottle is white and the liquid ranges from light to dark grey. We therefore assume that the background will always be darker than the bottles. Based on this, we can try to find the individual bottles by finding the vertical back stripes in the image. To do this, we can smooth the image to reduce the noise (e.g. with 3x3 or 5x5 box / Gaussian kernel) and then run a horizontal scanline towards the middle or bottom quarter of the image. Low values in the scanline correspond to the dark background; a significant rise will indicate the start of a bottle.

Find top and bottom of bottle neck & shoulder point.
Simply detect the top of the bottle itself. We assume that the perspective effect of the image is minimal for this image i.e. the bottles all appear the same size. After finding the top of the bottle, we can simply assume that the bottle neck and shoulder point are a fixed distance from the top. To find the top of the bottle, we can either search for the first strong horizontal gradient within the image, or use a scan-line approach,
similar to detecting the individual bottles.

Determining level of liquid.
Take a a vertical scan-line of the intensities down the (middle of the) bottle, similar to when we search for side of bottles, but this solution is less robust than searching for the gradient, since there are reflections in the area above the shoulder

Gabor Function Interpretation.
These knits feature predominantly vertical patterns of varying thickness. Therefore, it is okay to keep ω at a constant value to be oriented to vertical patterns. ϕ is the translation offset; since we convolve the kernel, it simply shifts the maximum to occur at the same offset; will not have any effect.
We want to vary λ to adjust the wavelength to distinguish the different vertical thicknesses λ often is set to be proportional to σ.

MOPS
MOPS will not work. The transformed image undergoes 3 types of transformations: scaling, rotation and inversion of the intensities. The MOPS descriptor is scale and rotation invariant but the descriptor values come from sampling the gray-scale values so once the greyscale values are inverted, it will no longer be able to match the keypoints.

SIFT
While SIFT descriptors are robust to such transformations, they are not invariant. A simple intuition is that projective transformations (and affine), with the non-uniform scaling in different directions will change local structures and shapes. This in turn affects the distribution of gradient orientations locally.

Why image doesn’t align well after DLT?
Images not related to each other linearly (DLT requires this)
Homography not applicable (not planar)

Why flow fields not accurate?
Aperture problem – edge regions in which resulting flow is perpendicular to edge

Briefly outline the design of your tracker, discussing how to initialize and update the filter and any assumptions that you make.
Target that we track are the tiles. Assume that we have tiles marked in first image, and a way to find new tiles when it first appears in the view. Initialize based on set of tiles observed in first image; update template when more (differently size, patterned, rotated) tiles appear on the conveyor belt; note we do
not update a given tile as it is tracked because it does not change in appearance

Number of parameters in MLP
$x \in 100 \times 1$
MLP A has 50 units in the first hidden layer and 10 units in the second hidden layer.
100 x 50 + 50 x 10 + 10 = 5 000 + 500 + 10 = 5510

Speeding Vehicle Detection
Motion segmentation via flow patterns; assumes cars do not drive too “close” in scene (this doesn’t work well in as it fades into the background)
- colour-based segmentation; assumes that cars do not have a similar colour as the street

How to improve tracker that cannot keep up & adjust for far away vs nearby camera?
Tracker does not adjust for scale
(1) Adjust the bounding box manually with some fixed percentage as it moves away from (decrease size) or towards (increase size) the camera
(2) Scale-sensitive mean shift tracker (add scale as one of the unknown parameters)
(3) Warp the scene so that the car size remains constant before applying the tracking algorithm