**Computer Vision: From Theory to Application**

Total 3 pages,

2018

Ph.D. Qualify Exam

1. (**20%**, Camera Model)
   1. If the camera sensor size is 1/3 inch with resolution *w\*h*=800\*600 pixels, based on 1.0X lens magnification (i.e., image resolution is 800\*600 pixels) please find the pixel size = \_\_\_\_\_ *um* \* \_\_\_\_\_*um*? **(10%)** (Hint: Please check Figure 1.)

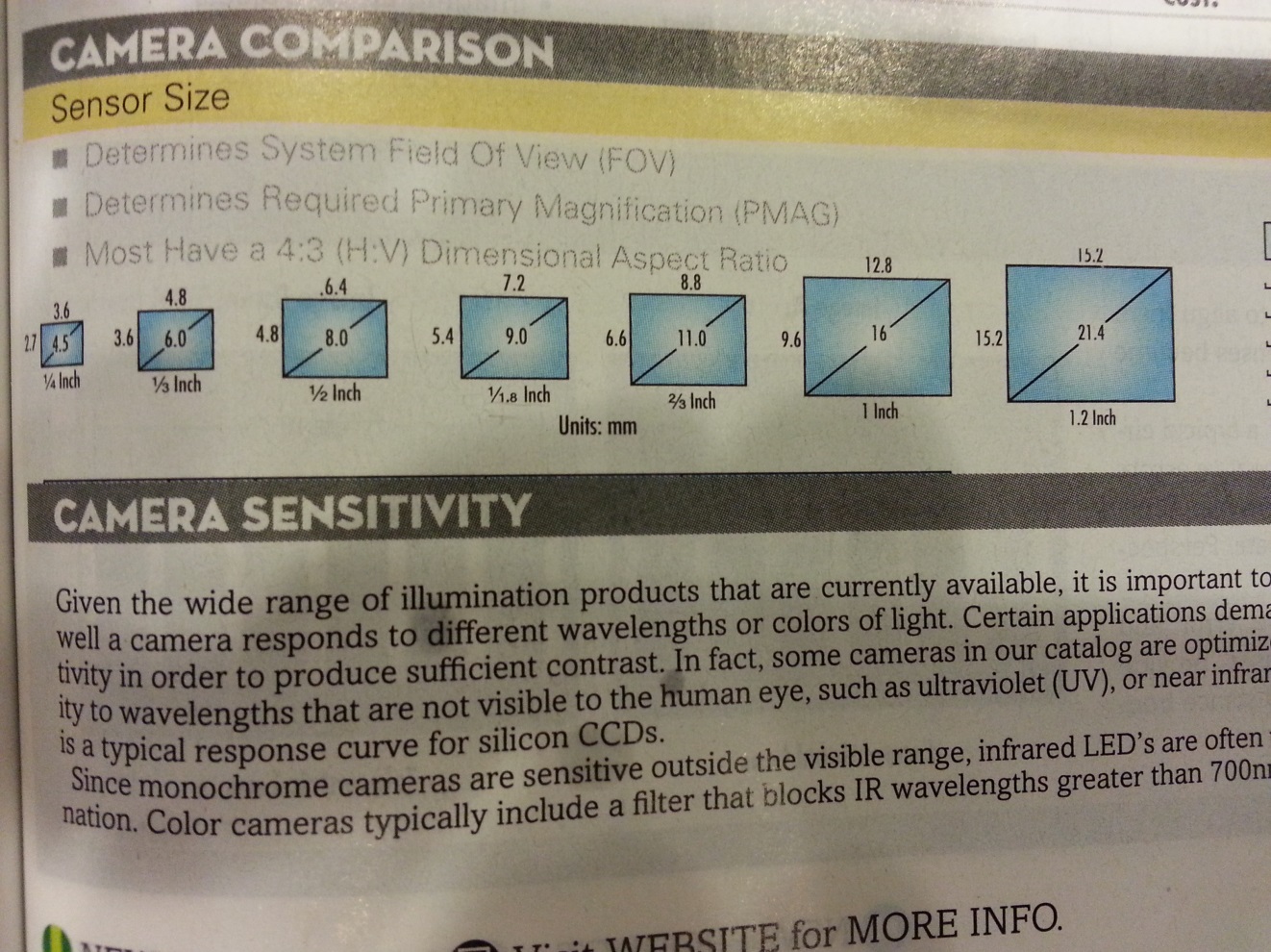


Figure 1

* 1. As shown in Figure 2, please find focal length *f* =function of (*w, θ*)? **(5%)** = ? *mm* **(5%)**



Virtual

Image Plane:

Width *w* (pixels)

Field of View (FOV) angle θ= 600

sin300 = 0.5

cos300 = 0.866

*w/2*

Pine Hole

Camera

(*x,y,f*)

Figure 2

1. (**20%**, PCA) For principal component analysis (PCA) computation, first the covariance matrix *C* is created. Second, the Singular Value Decomposition (SVD) is applied to *C* (i.e., *C* = *UDV*T) to obtain eigenvalue matrix *D* with corresponding eigenvector matrix *U*. Here, there are five data samples *x=*{*x1, x2, x3, x4, x5*} as shown in table 1 and corresponding low-dimensional (projected) data is represented by *y=*{*y1, y2, y3, y4, y5*}. Suppose the unsorted eigenvalues and corresponding eigenvectors of covariance matrix are shown in the table 2. If we want to reduce the dimension of data vector *x* from 5 to 2.
   1. What is the projection matrix *w*? **(4%)**
   2. What is the projection data (weight) vector {*y2, y3*}? (Please show all calculations) **(4%+4%)**
   3. PCA belongs to \_\_\_\_\_\_\_\_\_\_\_ model. (Hint: Gaussian, non-Gaussian, or Markov) **(4%)**
   4. PCA belongs to \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ learning. (Hint: Supervised, semi-supervised, or unsupervised, reinforcement) **(4%)**

|  |  |
| --- | --- |
| eigenvalues  Table2 | eigenvectors |
| 1 | (4, 3, 6, -2, 0) |
| 9 | (-3, 5, 0, 0 7) |
| 4 | (1, 8, -4, 5, 6) |
| 7 | (2, -1, 0, 4, 1) |
| 3 | (5, 5, 9, -1, 2) |

Table1

|  |  |
| --- | --- |
| *x1* | [1, 2, -7, 5, 3]T |
| *x2* | [4, -2, 6, 0, 3]T |
| *x3* | [-7, 5, -1, -2, 1]T |
| *x4* | [3, -3, 4, -2, -5]T |
| *x5* | [-1, -2, -2, -1, -2]T |

3. (**20%,** HMM) A HMM topology and parameters are as following graph.

0.3

1.0

0.2

0.3

0.6

0.5

1

2

3

π1=1.0

0.1

b1(O0)=0.6

b1(O1)=0.1

b1(O2)=0.2

b1(O3)=0.1

b2(O0)=0.1

b2(O1)=0.15

b2(O2)=0.05

b2(O3)=0.7

b3(O0)=0.3

b3(O1)=0.1

b3(O2)=0.4

b3(O3)=0.2

Please write its corresponding 1 parameter vector and 2 parameter matrixes, *π* (1x3 vector) **(6%),** *A* (3x3 matrix) **(7%),** and *B* (4x3 matrix) **(7%),** with values?

1. (**20%,** VQ) Please set the sort order for the procedure of Vector Quantization algorithm:











1. Set the sort order above 6 steps? (**12%)**
2. Which step includes the nearest neighbor rule? **(4%)**
3. Vector quantization belongs to \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ learning. (Hint: Supervised, semi-supervised, or unsupervised, reinforcement) **(4%)**
4. **(20%,** DL) Based on the deep learning lecture:
   1. For each neuron as following, if *ai* is the given (known) input image pixel, *wi* is the weight parameter of neural network, *b* is the bias parameter, and *z* is it output result. Please write its ***AX=B*** format? Here ***A*** is unknown parameter vector, ***X*** is the known input pixel vector and ***B*** is the output result. **(7%)** After activation function, the output of *σ(z)* is the \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ result **(3%)** (Hint: Linear Discrimination, Non-Linear Discrimination)



* 1. Each of following answers for physical meaning has only one answer selection:

(b.1) Deep learning, which is the same as AdaBoost, has the property of \_\_\_\_\_\_\_\_? **(2%)**

(b.2) Deep learning, which is the same as Supported Vector Machine, has the property of \_\_\_\_\_\_\_\_\_\_? **(2%)**

(b.3) Convolution process has the property of \_\_\_\_\_\_\_\_\_\_\_\_\_\_? **(2%)**

(b.4) Max Pooling has the property of \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_? **(2%)**

(b.5) Softmax function has the property of \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_? **(2%)**

(Hint: Non-linear discrimination, subsampling, feature extraction, cascade, output normalization**)**