Zhixing Chang

Computer Vision

Q1. True.

When a set of vectors are linearly independent, the result of coefficients from a1 to ai times vectors v1 to vi is zero, and all coefficients must be zero. When we have a zero vector, that means any coefficients which can be non-zeros times this zero vector will gave us the same result zero. So, when we have a zero vector, the coefficient can be non-zero so this must be linearly dependent.

Q2. True

When we have a set of linearly dependent vectors inside a larger set of vectors. Since the large set of vectors contains this set of linearly dependent vectors, then the large set of vectors is also linearly dependent.

Q3. Separability means a mxn matrix equals a 1xm matrix multiply nx1 matrix. This also means something similar to dimension reduction. When we separate a 2D matrix into two 1D matrix, that affect running time a lot. The mxn matrix takes mxn time to loop over but the two after separating matrix only takes m+n time to loop over.

Q4. Assume we have a matrix A. A is separable and equals multiplication of two 1D matrices B and C. So using image multiply A equals using image multiply B multiply C. From the question, the Matrix A is (2k+1)x(2k+1), Matrix B is 1x(2k+1), and matrix C is (2k+1)x1. So the number of operators for calculating with matrix A is NxNx(2k+1)x(2k+1). The number of operators for calculating with matrix B and matrix C is NxNx2x(2k+1)

The difference is : A- B&C

N^2x(2k+1)^2-N^2x2(2k+1)=N^2(4k^2+4k+1-4k-2)=N^2(4k^2-1)

Q5. Gaussian are more like center focused filter. So if we convolving a gaussian with another gaussian, the matrix we get should have much larger number at center than other numbers. From the border, the number rapidly increase to the center.

Q6. Dimensionality Reduction is to reduce the number of dimension for calculations. As we all know, colorful images are 3D matrix. If we transfer the image to 2D matrix to calculate, then we are using dimensionality reduction. The advantage is the program can run much faster. The running time for a 3D nxnxn matrix, if we try to loop over the matrix, it takes O(n^3). But if we are dealing with 2D nxn matrix, the running time should be O(n^2) which is much smaller than O(n^3). So fewer dimension means much much faster program. But there are bad things for reducing dimension. We may lose something from our matrix. For example, if we reduce the dimension of an image, from colorful 3D matrix to grayscale image with 2D matrix, we can calculate fast, but we lost all colors.

Q7.

 