*(Note: To add a line without triggering any stupid list features, use ‘shift-enter’)*

1. **Linear Systems**
2. **Make sure you understand what makes certain image operations linear and what are some operators we use in, say edge detection, that are not linear.** **Linear vs Non-linear.**  
   Linear operations are just different representations of the same image, such as blurring. Also, linearity maintains that the function holds under additivity (i.e., *F(a + b) = F(a) + F(b)*) and also homogeneity of the 1st order (i.e., *F(a\*x) = a\*F(x)*) Linear: convolution, correlation, smoothing filter, multiply sum are linear operations.

Non-linear operations fundamentally alter the image, such as thresholding. Median filter is edge preserving (gets rid of outliers), use that one in edge detection as opposed to a Gaussian filter which is linear but doesn’t preserve edges - (it blurs them) Median filter is not linear operator

1. **Describe how you might do edge detection using at least two operations - first a linear one followed by some number of nonlinear ones - that would find edges in a slightly noisy image.**

Edge Detection with linear then nonlinear.  
 Blur, 2nd derivative, then threshold and erode/dilate

1. **Gaussian vs. Salt and Pepper noise? Why does a linear filter work well to reduce the noise for the Gaussian case but not the other?**  
   Gaussian noise have values closer to the neighboring values. Gaussian noise is independent at each pixel and centered about zero. Because it replaces the pixel value by the local average, it’s fine when the noise is modest and tends to add to zero over a neighborhood

Salt and Pepper is straight black and white.

1. **How is sharpening done using filtering? And would it matter whether you used convolution or correlation?**  
   Double the value, subtract the blurred image

Convolution is correlation with the filter rotated 180 degrees, makes no difference if the filter is symmetric. It matters, not symmetric, shift image to left in convolution to right in correlation

1. **What are two ways to compute gradients in an image that has some noise in it?**

1. Smooth image via Gaussian kernel, then apply gradient filter (Sobel).

2. Apply gradient filter to Gaussian, apply to image.

Convolution and correlation

1. **What can you do during edge detection to account for the fact that some edges vary in contrast along the edge - that is sometimes thy are strong and sometimes weak.**

Enhance edge detection?  
Threshold and use hysteresis with high threshold to find strong edges, and a lower threshold to find edge points that continue those strong edges. (AKA Canny Edge Detection)

Using the Canny edge detector define two thresholds: low and high, use the high threshold to start edge curves and the low threshold to continue them

**2. Data Structures**

1. **A standard Hough transform performs voting for a parametric shape. Why are we doing voting and why does it work?**

It compares all points to all other points, and puts in a vote for every shape that can fit those points.

With a bit more detail, Hough transform takes feature points in an image and gets each feature to vote for a particular set of parameters that fit that feature, after each feature votes, the model that receives the most votes is a good fit for those features. Also, this works well since the majority of features will vote for the best model, and outlier votes will likely be in the minority and not affect which model is ultimately chosen.

Why doing voting: not feasible to check all possible models or all combinations of features by fitting a model to each possible subset.

How does it work: we let the features vote for all models that are compatible with it, it cycle through features, each casting votes for model parameters, then look for model parameters that receive a lot of votes.

1. **How can the Hough transform help in identifying lines, circles, and other shapes. You should be able to interpret a Hough Accumulator and determine what shapes are present along with details of each one (location, orientation, size, etc.)**

Review Accumulator arrays

Hough transform is a voting technique that can be used to each edge point votes for compatible lines, look for lines that get many votes. Let each edge point in image space vote for a set of possible parameters in Hough space.

Accumulate votes in discrete set of bins; parameters with the most votes indicate line in image space.

Line: polar representation for lines, point in image space is sinusoid segment in Hough space. Find values of (d, theta) where H[d, theta] is maximum. Detected line: d=xcos(theta)+ysin(theta)

Circles: center(a,b) radius r, for a fixed radius r, unknown gradient direction. Most votes for center occur in the center for each possible r possible direction theta, a = x – rcos(theta), b = y + sin(theta), H[a,b,r]+=1

1. **A friend needs to find the pool balls in an image of a pool table. Would a Hough transform be a good idea? Why/why not? Would RANSAC be better?**

RANSAC would not be a good choice, as it is intended to only find one solution/pool ball, not multiple. Hough would be a better choice as it can detect multiple objects. Pool balls are spheres, with well-defined edges, and should make Hough transform easier.

**3. Frequency**

1. **Fourier analysis decomposes images according to a basis set. What is that basis set?**  
   A series of sine waves
2. **How does the Fourier transform encode magnitude and phase of sinusoidal component of a signal?**  
   As a single complex number?

for every w from 0 to holds the amplitude A and phase f of the corresponding sinusoid

1. **Is the Fourier transform a linear operation? Why or why not?**  
   Yes, because the sum of two Fourier transform functions is the same as the Fourier transform of the sum of those functions. (another way of thinking about it is that it’s an integral and integrals are linear). Note that the *function* isn’t linear, but the *transform* is.
2. **Why does convolving an image with a Gaussian attenuate the high frequencies?**

(note for the English challenged: at·ten·u·ate verb əˈtenyəˌwāt/ reduce the force, effect, or value of.)

~~High frequency means high contrast -~~ ~~big gradient i.e edges~~. Gaussian is a low pass filter, which means it doesn’t let through the high frequencies. How does it do that? ~~By multiplying the high frequencies by a small number.~~  
edit. The above is unclear. I don’t think high contrast means big edges (It means strong edges). And I don’t see how multiplying something by a small number attenuates it (you get attenuation if that number is less than 1).  
 It’s a blur, right? Small details are usually lost when blurring

I don’t quite agree with the high frequency means high contrast as for the reason why high frequencies are attenuated by filtering. It’s really because we lost details when we blur, as noted right above my answer. We lose those details because when we convolve with a Gaussian, we are really also multiplying the transform of our image by a gaussian. Due to the shape of the Gaussian we multiply our spectrum by, we mainly only keep low frequencies and higher and higher frequencies get multiplied by a smaller and smaller values.

The real answer is in 2L-2C, which basically states that the Fourier transform of a gaussian is another gaussian. So when you apply a gaussian filter to an image you are convolving that image with the gaussian. When you switch this over to the frequency domain that convolution becomes a multiplication. Now you have the natural frequency profile of the image being multiplied by the gaussian, which will attenuate the outer frequencies.

FT of a Gaussian is also a Gaussian, FT of a convolution is a product. FT of the convolution of an image with a Gaussian is the product of FT of this image with a Gaussian, which has its higher frequency multiplied by a small number.

1. **What is aliasing and when does it happen?** **Draw a picture that explains it in terms of a comb filter doing the sampling and the effect of that operation in the frequency domain.**  
   Any loss of information, usually from sampling.  
   Not quite accurate. Aliasing is when we are undersampling a signal, and since we are undersampling, frequencies that are higher than we can possibly sample become represented by lower frequencies than what is really there. So our sampled signal shows a lower frequency than what it really should be. Adding onto that, can be interpreted as signals in the frequency domain summing up. What’s added together cannot be split back up into its constituents.
2. **What is a gaussian pyramid and why is it so cool? What is the relation between a Gaussian pyramid and aliasing? In particular, why can you reduce the size at each step and hardly lose any information?**You can create a gaussian pyramid by blurring and then subsampling an image. When reducing the size at each step, the blurring effectively allows some of the information from adjacent pixels to “bleed” into each other. When we throw away pixels through downsampling, some of that information (i.e., high frequency information) is retained since the gaussian spread out that information a bit, so it isn’t effectively completely lost.

**4. CAMERA MODELS AND CALIBRATION**

1. **What is aperture? Why would you want large or small?**  
   Aperture is your ‘view’ or sample of the image. Aperture is a small hole through which light is directed.

Large Aperture means more light, but the picture is blurry, decreases diffraction. Small aperture means the picture is very sharp (focused) but with little light, provides a large depth of field where you can have whole image in focus.

1. **What is depth of field vs aperture**Depth of field is how in focus near or far objects are. It is directly related to camera aperture, ie. changing the camera aperture changes the depth of field. The aperture controls depth of field, and they are inversely proportional, i.e., wider aperture = shallower depth of field, and vice versa.
2. **Zooming vs moving closer**They are not the same. If you zoom, it is still difficult to tell the distances between objects, but the objects stay in focus. Zooming changes the focal length to match the distance of the object, but at a cost of limiting field of view. Zooming changes the angle of view while keeping the spatial relationship between the camera and the objects in the image the same.

If you move closer, distances between objects become more exaggerated, and field of view becomes smaller, which can be good or bad, depending if you want certain things in focus and other things appearing out of focus. This is a consequence of the fact that if you move very close to something and take a picture, it may become closer to you than the focal length, thus resulting in some perspective distortion.

1. **Perspective projection: A point in 3D at location <X,Y,Z> in the cameras coordinate system appears where in the image? And, what assumptions about the intrinsics did you just make?**

It assumes normal Z negative where d is distance from origin (focal point) to the image plane.

1. **Why do lines converge at the horizon?**It’s just a natural phenomenon? Because things farther away appear smaller?

Because those lines are perpendicular to the picture plane, and converge in the vanishing point.

Sets of parallel lines on the same plane lead to collinear vanishing points

1. **How many degrees of freedom are in the extrinsics and intrinsics? What are they?**extrinsics: 6 degrees of freedom: 3 for rotational position (yaw, pitch, roll), and 3 for translational axes.

6, fix one point, 3 DOF, fix second point, 2 more DOF (maintain distance constraint), third point, add 1 more DOF

Intrinsics: 5 (pixel sizes - width, height; center offset - in x and y; skew)

5, a focal length, a pixel x size, a pixel y size, two offsets and a skew, and f always multiplies the pixel sizes, those 3 numbers are only 2 DOFs

1. **How many 3D points need to be observed to do absolute calibration? Why?**

6. There are 11 total degrees of freedom, and each point gives us the answer/equation for 2 degrees of freedom, (one for x, one for y). 11/2 = 6

minimize solve for unit vector of with smallest eigenvalue, given 3D to 2D point correspondences , determine the “maximum likelihood estimation” of M

1. **Write the perspective projection equation as a 3x1 = [3x4] \* [4x1] How many unknowns are in the above equation?**

11 unknowns, homographies and 3D plances

1. **One way to solve for the unknowns is to view some points whose 3D position is known and whose 2D position is recorded. How many equations do I get per viewed world point? If I have, say, 10 points, how would I solve for those unknowns.**

Each point pair gives two equations, 10 points need 20 or more equations, the more the better

Solve the system of least squares of

[ x y z 1 0 0 0 0 -ux -uy -uz -u] \* [m00] = 0

[0 0 0 0 x y z 1 -vx -vy -vz -v ] [...m23]

Where u, v are your points.

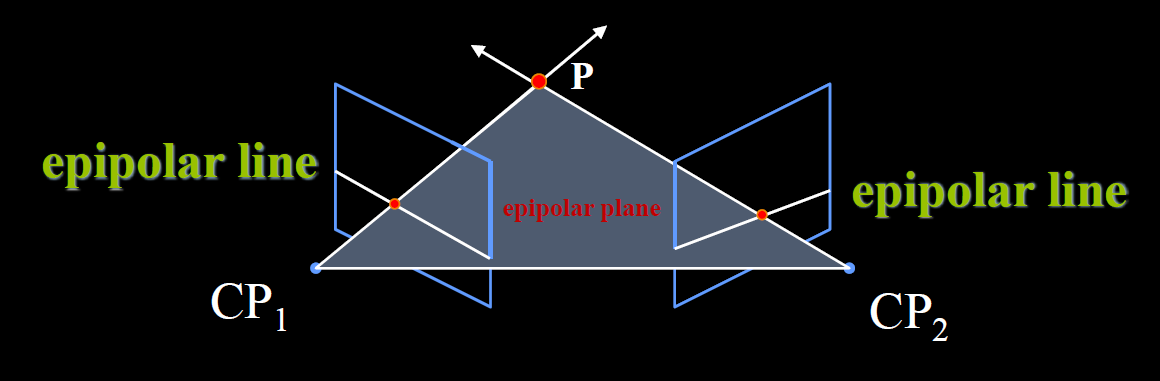
**5. N-Views**

1. **What is affine transform? How many pairs of matching points between two images does it need?**  
   A transformation where parallel lines remain parallel (rotation, scale) Affine transform is a linear mapping method that preserves parallel lines, ratio of areas  
   3 pairs? Yes 3 pairs (preserves rotation, translation, scale and sheer)
2. **What is homography? How many pairs?**A ‘projective transformation’ where the data/image is projected onto another plane. Lines remain lines.  
   4 pairs.
3. **Rectify a plane – i.e. why you can convert the image a slanted plane such as the face of a building into an image of building as if you were viewing it head-on.**

Mapping between planes is a homography. Whether a plane in the world to the image or between image planes. Measurements on planes: unwarp then measure if a planar rectangular grid in the scene, you can map it into a rectangular grid in the image.

**6. Stereo**

1. **Draw the epipole geometry**

****

1. **What is an epipole?**  
   The intersection of the optical centers of two cameras. Actually, the line connecting the optical centers of the two cameras is called the baseline. Whatever that means.

point of intersection of baseline with image plane

1. **Essential vs fundamental**  
   Each matrix describes how to convert a point in one image to a point in another image.

Essential assumes the cameras are calibrated, and converts a point to a point. Essential has 5 data points,

fundamental has 7 data points? Fundamental assumes cameras are not calibrated, thus it is impossible to find the exact point, so it is able to convert a point to a line the matching point is on.

Fundamental matrix is more general form than essential matrix: removes the need to know intrinsic parameters if estimate fundamental matrix from correspondences in pixel coordinates, can reconstruct epipolar geometry without intrinsic or extrinsic parameters

1. **Write the formula for the depth *Z* of P in terms of *d,* B, and *f*.**  
   Z= f \* (B / d) where d = xl – xr

is the disparity

1. **What are some constraints about the viewed surface or that matching that reduce the search in looking for stereo matches?**  
   Epipolar constraint, Epipolar constraint reduces the correspondence problem to a 1D search along an epipolar line

Soft: similarity, uniqueness, and ordering. Ordering constraint. The surface is planar

Hard constraints: the line containing the center of projection and the point P in the left image must project to a line in the right image. Corresponding pixel for some image point in the first view must occur in the second view

1. **What’s the difference between normalized correlation and regular (cross) correlation?**   
   Correlation is sensitive to photometric changes.  
   Normalized correlation is sensitive to non-linear photometric changes and even slight geometric ones. Normalized correlation subtracts the mean image and divides by its standard deviation. This is used to remove lighting/illuminance, making it insensitive to changes in lighting.
2. **What does random dot stereograms tells us about human stereopsis?**  
   It tells us that human vision processes focusing independent of any shape, color, or any data at all.

Random-dot stereograms highlighted a problem of stereopsis, the correspondence problem. This is that any dot in one half image can realistically be paired with many same colored dots in the other half images

**7. Shading**

1. **What is Lambertian shading?** **And what does it say is the relation between the incident light angle, the normal, the viewing direction and brightness?**

Lambertian shading is where the shading gets darker the greater the angle between the surface and the light source.

It says the incident light source is proportional to the darkness of the color. Viewing angle does not change (You can view it from any angle and will not see a change in darkness or lightness). Viewing angle and such do not change the apparent brightness of the reflectance, even though more light is reflected back from perpendicular directions and drops off as viewing angle is shallower, but due to foreshortening the brightness appears to remain the same.

Brightness: The more perpendicular the light source is to the surface plane (the normal), the greater the light.Lambertian shading regards a BRDF model in which a surface reflects light, and hence it’s apparent radiance (brightness) in all directions, is equal.

1. **how many known light sources would you need to turn on (one at a time) to unambiguously figure out the orientation of the surface?**

3?

1. **What are the 3 degrees of freedom?**

X, Y, and angle? I believe it should be albedo, source vector, and orientation normal vector (what we’re looking for)

reflectance aka albedo

angle between lights and n, three directions to the light source

I illuminance (strength of lights)

**8. Features**

1. **What is invariant and distinctive. Why is it good?**  
   Invariant means the feature is not affected by change in illumination or rotation. Invariant point descriptors means they are almost the same in the two images. Found each time. Invariant means it appears the same in any image, even if it is moved. It means you can find it again. They are good because when identifying images, you need to locate points that exist in both images.

Distinctive means it is well defined in both dimensions. Distinctive descriptors means you can tell them apart, unambiguous. You need to find matches, and make sure you are not getting false matches. (ie. a line would result in many matches, as every line looks much like another. Corners are better, as they are well defined and you can get their exact location)

Invariance and distinctive are a trade-off. The more invariant, the less distinctive it is, and vice versa

1. **Harris features are referred to as “Harris Corners” and are found by looking at a 2nd moment matrix. Why and why? And what does it mean if the largest eigenvalues of that matrix is much, much, much bigger than the second one?**

A second moment matrix is the product of the derivatives.M = ATAis the second moment matrix, the eigenvectors and eigenvalues of M relate to edge direction and magnitude, classification of image points using eigenvalues of M

When one eigenvalue is much bigger than the other it means you are along an edge. When one eigenvalue is much bigger than the other it means you are along an edge, not a corner.

1. **How can we make a feature detector (like SIFT) mostly invariant to illumination?**

Normalize the 128 number feature vector to magnitude 1.0

Clip gradient magnitudes to avoid excessive influence of high gradients.

After rotation, normalization, clamp gradient >0.2

1. **Are Harris corners invariant to rotation? Why or why not?**

Yes, when you compute the 2nd moment matrix and R, the Harris corner response function, you get the same shaped ellipse, the same eigenvalues, the same R

Because harris corners only look for eigenvalues of a region - they do not care what direction or angle those eigenvalues are, only that two of them exist in about equal strength.

1. **Are SIFT features invariant to rotation? Why or why not?**Yes.

By creating a histogram of local gradient directions. Histogram could be flipped, but the dominant direction will still show up as the peak. Part of SIFT features are to find the dominant direction, and rotate them so they all point up. Image content is represented by a constellation of local features that are invariant to translation, rotation, scale, and other image parameters

1. **Are Harris corners invariant to scale?**

No, when you zoom into a corner, its features look like edges, not a corner any more

**9. Model Fitting**

1. **In using RANSAC to do, say, a panorama, what are putative matches? How do you get them? Why do you need them?**

Putative just means possible, or potential, matches. They are our ‘preliminary’ matches.

They are some plausible matches, well which features are likely to match which other features. Might get some mistakes but at least some reasonable matches doing this.

How to get: hypothesize some transform T from some the matches, verify loop until happy. Taking different matches, calculate the SSD, less than a threshold, accept that/doing exhaustive search, pick the one best matching by comparing There are many ways to get them. Nearest Neighbors, wavelet matching, SIFT, K-d trees, best bin, hashing…one against every other, also hashing and nearest neighbor techniques.

Why need: You need them to test different hypothesis, on which transformation leads to the most matches. kd-tree or the best bin, figure out a solution to ignore the wrong and pay attention to another.

1. **Suppose we are using RANSAC to find circles. Our inputs might be points or oriented edge elements. What would the argument be as to why points are better? What would the argument be as to why the oriented edge elements would be better?**

Points better: if just looking one or two circles with partial overlap, in low noise conditions, making algorithm more simple and fast. The RANSAC algorithm, by definition, uses points. It matches points to shape. With oriented edge elements, (unless the edges are already perfectly aligned with their shape) it is not clear what is a match or not.

Edge elements: if circles overlap, edge orientation to determine which circle the points belong to and in presence of a lot of noise, use the edge orientation to determine whether a point could belong to a circle, if not, discard it to obtain a closer fit. However, if you modified the algorithm, and edges are perfectly aligned, this would lead to a reduced set of points to search, since one edge, normally many points, could be represented by a single point.

**10. Segmentation**

1. **How can segmentation be thought of as a clustering problem? How do you get geometry into that approach?**

Segmentation is separating an image (or any data) into related parts. Clustering is the method used to do segmentation. Clustering IS segmentation. It is the manifestation, the actual nuts and bolts, the algorithm, used to do segmentation. Clustering techniques are used to group data/observations in a few segments so that data within any segment are similar while data across segments are different, carve the histogram into different groups, the histogram of intensity of pixels.

Most clustering algorithms are geometry-based, meaning they use Euclidean distances to determine groups. Finding the cluster centers, best clusters whose centers minimize the SSD, the cluster is a distribution that’s centered about a point with a Gaussian fall off around it. Some value whose probability falls off as the square of the distance. The value that would generate the maximum likelihood of the data is the mean value

1. **What does Mean Shift do and how does it relate to segmentation?**

Mean shift is used to find the centers of distributions, by taking a point and its surrounding region, and moving that point closer to the center of mass of that region. When it no longer moves it has reached the center of the cluster/distribution. It is how we can determine the ‘K’ in K-means clustering. It can also be parameterized to specify certain attributes (ie. shape, appearance) of your clusters.

The mean shift algorithm seeks modes or local maxima of density in the feature space

Cluster: all data points in the attraction basin of a mode. Attraction basin: the region for which all trajectories lead to the same mode. Find features initialize windows at individual feature points (pixels) perform mean shift for each window until converge, merge windows that end up near the same peak or mode

**11. Motion**

1. **What is the Brightness constancy constraint equation and what are the unknowns?***I* (*x*, *y*, *t*) = *I* (*x+* *u*, *y+* *v*, *t+*1)  
   u, v are the unknowns.

Edit - This assumes delta t = 1

The full equation works for any length of time

I(x,y,t) = I(x + change in x, y + change in y, t + change in t)

This says the next image is just a previous image + some changes.

1. **What is the aperture problem in considering image motion?**We can only tell the amount of motion that is perpendicular to the edge in a little area. When you think about this little area, you can think of it like looking through a little hole from cameras, that's called an aperture. And this general problem is called the aperture problem.

The aperture problem is that if you don’t have a large enough region (ie. aperture) around your feature, when tracking the feature, it may appear to move in one direction, but is actually part of a larger whole that is moving in some other direction. The solution is to use better features, like corners.

1. **What is the relation between the Lucas and Kanade optic flow method and finding the Harris corners**

Both need the area in question to be distinctive, ie. not a line, ie. two strong eigenvalues.

Solve aperture problems, compose some form of constrains, local constrains of Lucas and Kanade. Having local constrains could have more than one equation per pixel, one pixel with multiple equations, gave us least square solution, related that to the whole Harris corners. Matrix: two decent sized eigenvalues it meant that the gradient occurred in various directions over that window. Fitting a planar ramp, the first -order derivative approximation.

1. **Lucas and Kanade is the optic flow method based upon gradients. What are the assumptions of the method? And what can be done to apply the algorithm when those assumptions are false?**

Color constancy: small motion, pixel hasn’t moved didn’t change its color, didn’t change its intensity. The 2 assumptions are that movements are generally small and constant within a general area. They do not randomly appear far away.

I have no idea what to do if these assumptions are wrong. Increase the scale? Look at the big picture, not small patches? To do something in terms of feature detection, SIFT detections and characterization. Also, motion segmentation. If the main assumption, that is movement is small, is violated, then we use hierarchical LK with a gaussian pyramid. This is in essence changing scale, so as to reduce resolution to a point where those large movements are really just small movements with regard to how many pixels objects have moved.

1. **How would you work the knowledge that there is affine flow only into the LK method?**

Reduce the scale.?

This is done by instead of assuming it’s the same <u, v> over a window, we assume it’s the same affine transformation model represented by <u, v>, this allows for larger windows (since instead of assuming they’re all moving the same, they’re actually moving the same kind of way following an affine transform)

The foreground is moving in some way not a constant then it’s can affine flow. It’s a planar. The background may be moving some other way. To find that, do a local flow estimate.

LK gives vectors, see big jumps from those values break that up into segments to see which segments seem to be along the same plane. It’s planar because of affine equation, fitting the affine plane.

**12. Tracking**

1. **Tracking is iterating between Prediction and Correction. In terms of the observations, prediction can be written as:  
   Screen Shot 2015-04-24 at 11.45.51 AM.png  
   Write out a similar expression for the correction step.**
2. **In such tracking what is the role of the dynamics model? The likelihood (observation) model?**   
   dynamics model: independence assumption: want the probability of Xt, given the previous state, the only thing matters, belief: everything up through t minus 1 dynamics. The dynamics model predicts or estimates the next position.

Observation model: measurements make use of this observation model, the likelihood of the measurements depends only on the current state. The likelihood estimates the probabilities of each measurement given some position? It is a weighted average of the measurements for each position? The first option for likelihood given here is correct, it is the probability of some measurement given our predicted location (after updating with dynamics model)

1. **There are two independence (or conditional independence) assumptions in the tracking we did (Kalman or Particle). What are they? Hint – one has to do with the states, the other with the observations.**

Markovian: only the immediate past matters. Past and future states are dependent on the present state.

Measurements depend only on the current state, assume that objects don’t disappear and reappear instantly somewhere else, the camera doesn’t move instantaneously

1. **The Kalman filter imposes Gaussian distributions for the state estimation and two other model elements. What are those elements?**

The covariance of measurement noise and process noise.

Prediction, correction, processing

1. **Particle filters first sample from a weighted distribution of particle, each particle being representative of the state. After that sample is picked, what is done to the sample *before considering the measurements***control and diffusion, sample the new state using whatever the old one was. Action sample from new distribution. Noise, generate a little bit of noise and add that. Sampling: previous sample, control, the noise, reweight by how likely the measurements would be.

Sample

*Get movement from sample, apply that movement, estimate its probability or likelihood (aka predict)*

*Multiply that prediction by its likelihood*

*Multiply that by the likelihood of the measurement given its state*

Resample

Normalize

repeat

**13. Classification**

1. **If we reduce the number of dimensions of a signal using PCA, we first subtract off the mean. Why?**  
   In general, there are M minus 1 eigenvectors. And the reason there's M minus 1 is, I take my endpoints and I subtract out the mean. So that removes a degree of freedom. So that's a trick question that I always give on the final of my course. So if anybody's taking the final, now you know the answer. Okay, there's M minus 1.

Edit: really? Where are you getting this? And it doesn’t answer the question.

The mathematical definition of PCA is defined with respect to the origin. Subtracting off the mean is how we center our distribution/data at the origin.

Both answers are technically correct (the original answer given above is apparently directly copied from the lecture subtitles). We subtract off the mean to center our distribution at the origin, but a consequence of subtracting off the mean is that when we find the eigenvectors for our MxM matrix, we don’t get M eigenvectors, but we get M - 1 since subtracting the mean removes one degree of freedom.

We are doing variation about the mean, so we subtract the mean and take dot product and U of Xi would be the coefficient. Find the u that captures the most of variance.

1. **What’s the difference between generative models and discriminative models for classification? Which relies on Bayes rule and how?**

generative model: pure probability allows to use priors about each category, just build a model for that category **but** Discriminative model: worry about a lot different categories, also use it to generate data, not really leveraging having large data. Generative models train on data to describe what fits in a particular class, with a goal to determine to which class certain data belong. Generative says what something IS. Generative uses Bayes rule, because Generative usually must pick the most probable match, given some data. Dealing with probabilities is the domain of Bayes. Generative models rely on Bayes rule to determine P(class | x ) from only using data (since we can get our likelihood and model our prior from data).

Discriminative model: model the hard case tight similarity between them, modeling lots of data,

Discriminative models, however, do not train to model the classes themselves, but the separation boundary BETWEEN classes. Discriminative says what something is NOT.

1. **What’s a *cascade (filter)* and how is it used with boosting for face detection?**Even if the filters are fast to compute, each new image has a lot of possible windows to search almost everywhere is a non-face. So detect non-faces more quickly than faces. And if you say it’s not a face, be sure and move on.

The cascade filter is the process of singling out false positives during training, and selecting the Haar Feature that generates that false positive, removes it from the matching set, and repeats. It stores these false positive features and uses them separately, in order, and if a feature is not detected using these false positive features, you can be certain it is not a match, and we don’t need to continue through the rest of the other false positive features.

1. **What are integral images and why are they so useful?**The value at (x,y) is sum of pixels above and to the left of (x,y)  
   sum = A – B – C + D  
   It’s running fast, more efficient.

Integral images are a preprocessing step where all the sums of a region are computed so that the point x,y is the sum of all the points above and to the left of x, y. In this way we can compute sums of sub-regions in 4 additions, ie. constant time. It is unaffected by scale and makes calculating viola-jones features very fast.

1. **What is the *Kernel trick*? And how do we make use of it with SVMs?**

A kernel function is any function that corresponds to a dot product in higher dimensional space. These functions therefore are dot products, and can be used as a linear classifier in our SVM.

A kernel function is a similarity function that is the dot product (inner product) of the higher dimensional space it can be used in the linear classifier. We don’t actually need to know that space. Use the kernel function in SVM, its dot product in high dimensional space, don’t even look at it. Convert XiX to kernel, don’t need to do things in the high dimensional space, can compute kernel in low dimensional space.

1. **How do we define the “bag of words” that is used for recognition?**

Generally we take the most common features. How you define common features depends on the method used to find features.

Describe the entire image by the histogram or the collection of those words. Took all images together, find all the visual words, gives me a vocabulary. Build histogram for each of the objects. Train a support vector machine to make the decision. Don’t know parts, structures, build description, word-like collection of those descriptors. Find the descriptors that are present referred to as bag of words.

**14. Activity**

1. **An HMM is defined by a triple written in class as (A, B, pi) but in the book as (P, Q, pi). What is each of these? (Or “What are the three elements that make up an HMM?” If you can’t remember which is which)**A- state transition probability  
   B - emission probability  
   Pi - Initial state distribution
2. **What are the three fundamental problems to be solved when using an HM? And what is the forward algorithm?**
   1. Given the model lambda, what is the probability of occurrence of a particular observation sequence
   2. Decoding: optimal state sequence to produce an observation sequence
   3. Learning: determine model lambda, given a training set of observations

Forward algorithm: instead of looking at all possible paths, by using recursion, look at n squared t paths. Formula only looks back at the previous state, taking advantage of Markovian property. Evaluate the probability of seeing that entire observation sequence. The foward algorithm is explained in the below question (weird). It is called forward, because it creates an extra variable, alpha, which looks forward ie. time step t. It is the probability of a sequence at time t, assuming some state.

1. **If N is the number of states and T is the number of observations (one per time step), the forward algorithm gives a recursive method of computing the probability of given HMM producing the observation sequence (written as P(O | ג) ). What is the computational complexity of that computation in terms of N and T?**computational complexity = O(N^2 T)

As opposed to TN^T which is the complexity for the forward algorithm.

1. **And just how are HMMs used in activity recognition?**

Detecting gestures. The emissions of the HMM are the direction of the movement of the hand in a plane, and the sequences of those directions are unique for each gesture. So given a sequence, guess its state.

Use HMM to output the probability or the likelihood of getting a particular movement for a given action. Think of the entire contour as a sequence of discrete steps. Sequence of discrete symbols. HMMs: Train HMM each action will be an HMM, hookup multiple HMMs together

**15. Morphology**

1. **How are OPEN and CLOSE defined in terms of Dilate and Erode?**

Open means erode then dilate  
 Close means dilate then erode

1. **What is the effect of using a bigger structuring element when doing a CLOSE operation as opposed to a smaller one?**

The changes will be larger, and you will lose larger blocks of pixels that stand alone!

Close fills in pixels, meaning you will lose larger blocks of dark space, meaning you can fill in larger holes, or black space, Within shapes.

1. **What is the minimum number of repeated CLOSE operations you can perform until you stop seeing any further effects?**

One? Yes just one, since it is idempotent, meaning repeated operations won’t do anything the first one didn’t do.