**Localization**

Trying to find where we are in the world.

GPS has 10m of accuracy, which is far too inaccurate for a car.

We want at least 10cm of accuracy for a vehicle.

**Total Probability**

Let’s say there is a 2D world where a robot can move left to right.

The robot doesn’t know where it is. It has a map, of that is all the places the robot can be, vs the vertical axis of probability.

This function can map the probability of a robot being in a given location.

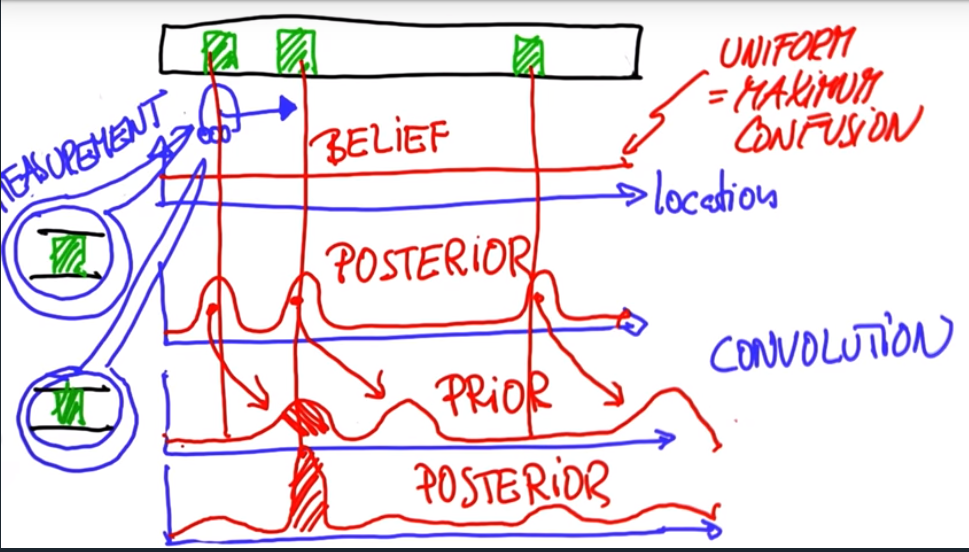
We can then model the robots current belief of where it is in the world by a Uniform = Maximum Confusion, meaning that the robot doesn’t know where it is, and every location we have mapped has an equal probability of being the location of the robot.

What the robot can do, is look at the features of the world, like whether it is in front of a door or not.

What if a robot sees a door? Our belief function transforms to have a higher probability at the locations with doors. The resulting function (Probability Density) is called the Posterior Belief (Meaning it is after a Measurement).

Now lets say a robot moves, to the right, the current belief state would then shift the same distance that we moved, which is not a 100% accurate measurement. This will lead to the shifted belief having a little bit of a lower peak than before the movement. The process of moving the beliefs to the right is called a convolution.

If we then measure another door, the belief transforms again to a high belief being in an accurate location.



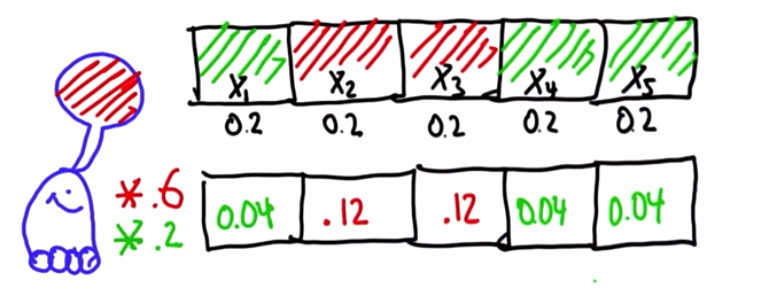
Example:

There are 5 blocks, 2 red and 3 green.

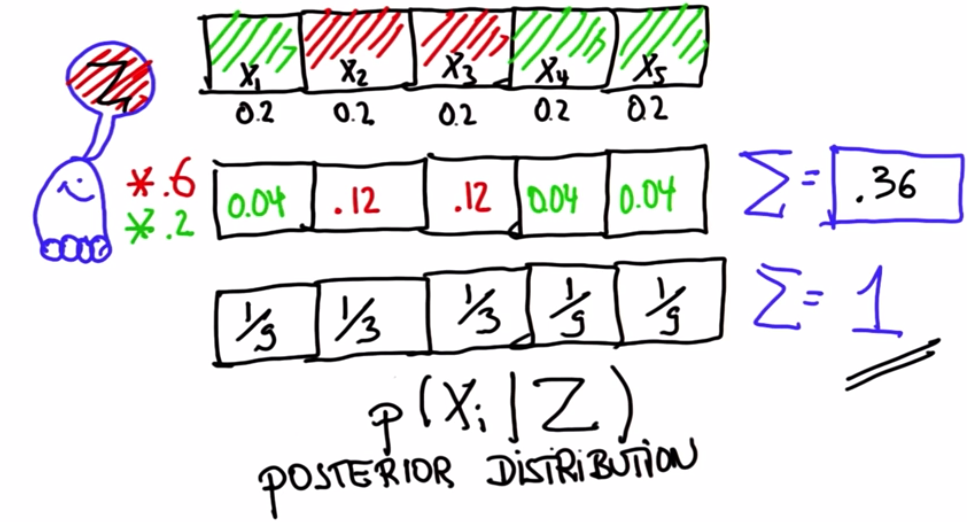
A robot sees that it is in a red block. This would make the probability increase in each of the red boxes, while decreasing in the green so that the total probability is still 1.

We can get the resulting distribution by weighing the red blocks more through multiplication.

Let’s say red blocks get multiplied by 0.6, and green by 0.2. The ratio shows that because we see a red block, the likelihood that we are in a red block goes up by a factor of 3.



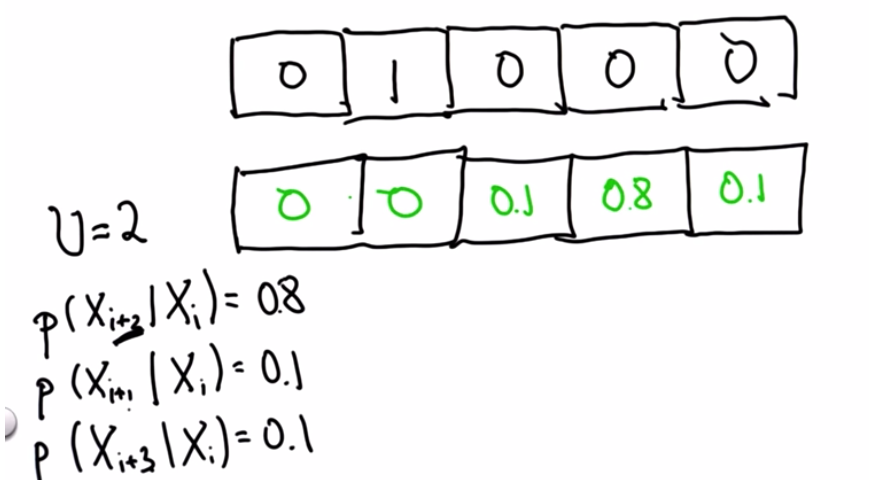
We then need to normalize the values so that the total probability stays as 1.

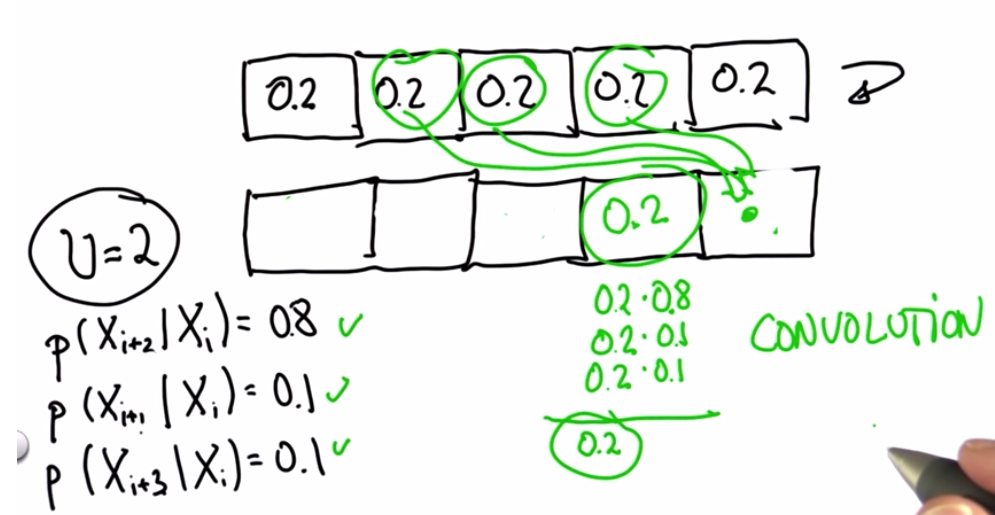


**Inaccurate Robot Motion**

If a robot had the probability that it landed in the exact estimated position is 0.8, and 0.1 that it undershoot and overshoot its estimate.

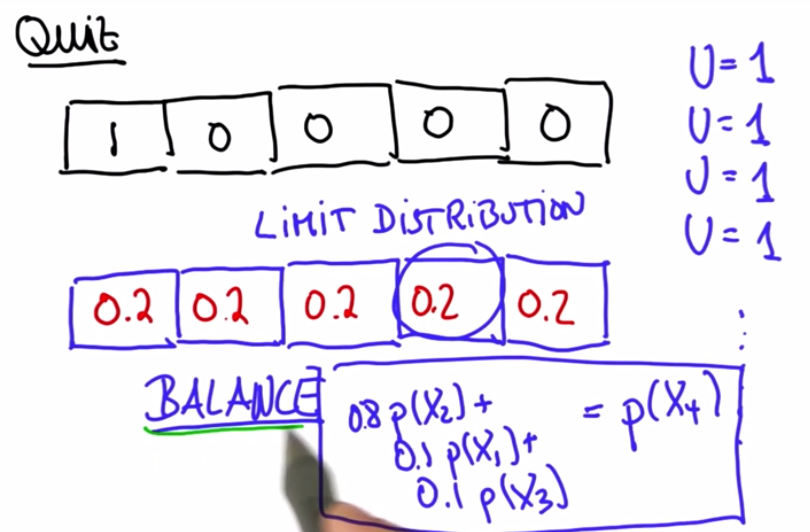
This is very important because robots are not 100% accurate.





Limit Distribution:

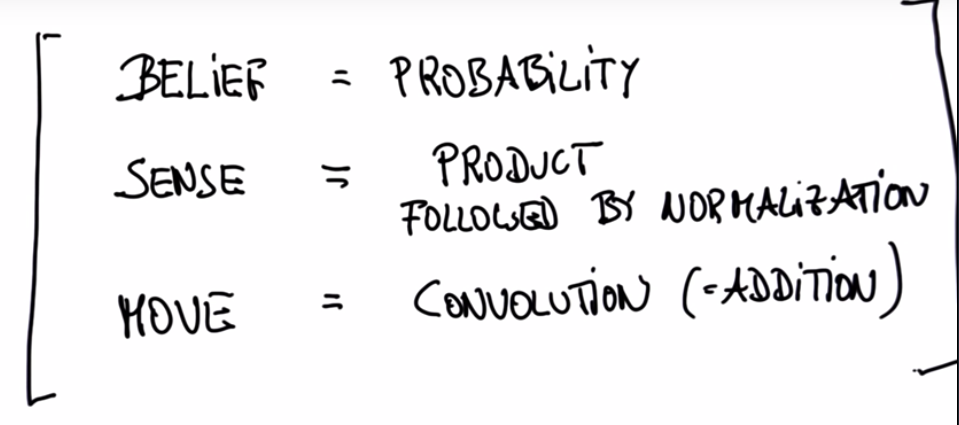
If we move an infinite number of times, the limit of the probability distribution is the Maximum Confusion case.



Look at Code Examples:

Measurement of information loss and gain is called Entropy.

Google self-driving car uses a camera image and measures the color difference between the lane and the road, and that is used to localize the vehicle.



**Bayes Rule**

