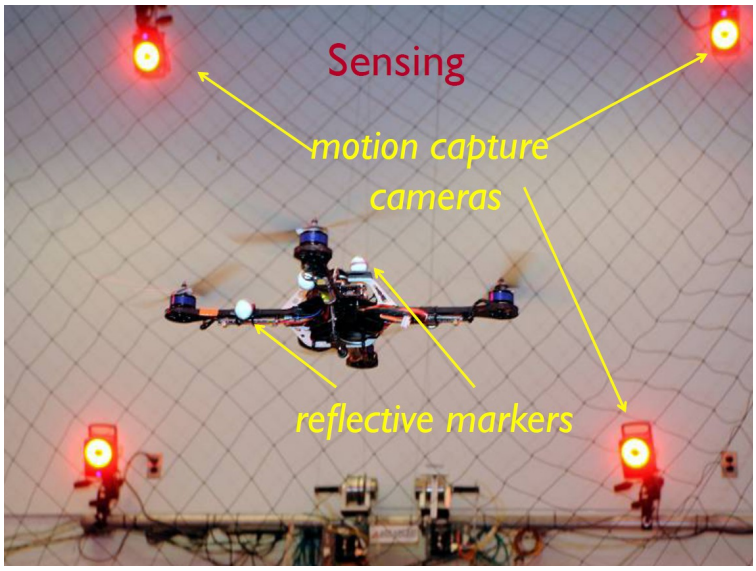
*Lecture 4.1*

# Sensing and Estimation

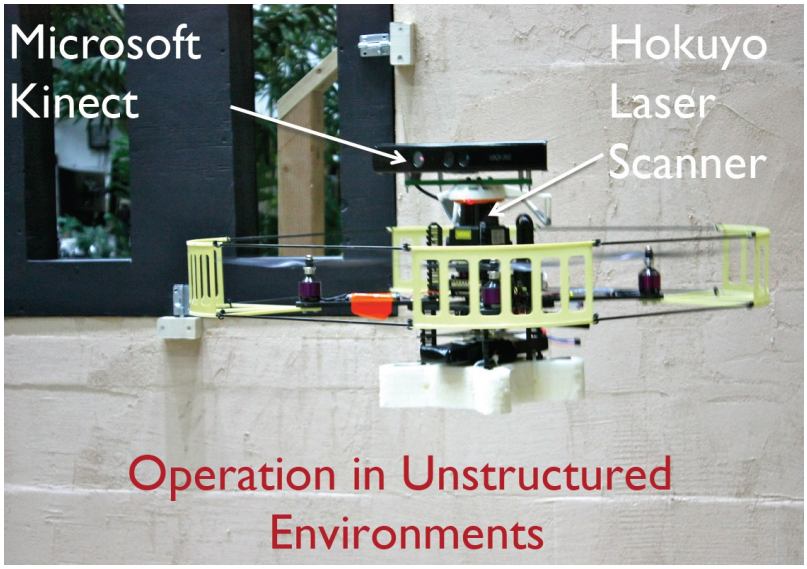
We're now going to turn our attention from control and planning to sensing and estimation. In everything we've seen so far, we've conducted experiments with quadrotors in a laboratory environment. This laboratory environment has motion-capture cameras that are able to measure different features in the environment with precision and repeatability. We can have reflective markers on quadrotors, allowing the cameras to estimate the position of a quadrotor to within several millimetres, and the orientation of a quadrotor to within a fraction of a degree.




Position updates can be obtained at rates as high as 200 hertz, so, in a sense, what we have is an indoor 'GPS-like' system, but with a degree of precision that's several orders of magnitude higher, and with the ability to get the information much faster.

But we want to operate in environments where there are no motion-capture camera systems. Robots must have the ability to estimate their state, their position, their orientation, and their velocities. And they must be able to do it onboard. This is the problem of ***onboard state-estimation***.

We can do this by equipping a robot with camera-like sensors. In the example below, the robot has been equipped with a laser-scanner, enabling it to measure distances to obstacles in the environment. It is also equipped with a RGB-D 'Kinect' camera from the Microsoft XBox system. The Kinect sensor projects infrared patterns into the environment and then observes how those patterns are deformed by three-dimensional features:



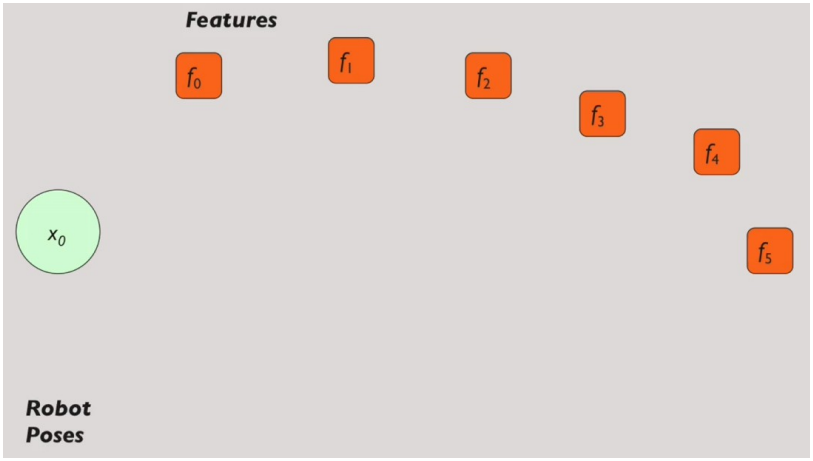

Both the laser scanner and the Kinect sensor allow the robot to measure three-dimensional features in the environment. This allows the vehicle to operate in unstructured environments without a motion capture camera system.

Here is another example of a vehicle that is able to operate both indoors and outdoors. It has a GPS sensor on the top, as well as two-forward facing cameras (and a downward facing camera that cannot be seen in this picture). The robot also has a laser scanner and is equipped with an onboard inertial measurement unit (IMU):

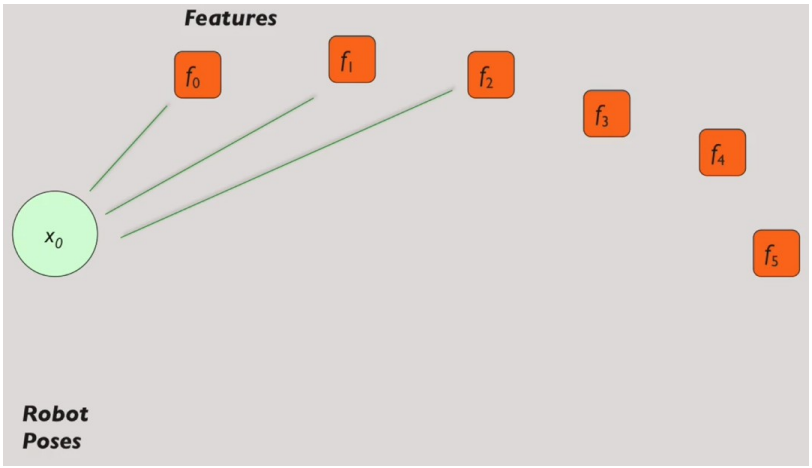

With all these sensors, the robot is able to track three-dimensional features in its environment, and use this information to track its own position and orientation as it flies through that environment.

The basic technique is called ***Simultaneous Localization And Mapping*** (SLAM), sometimes also called ***Structure From Motion***. Here's the basic problem. Imagine we

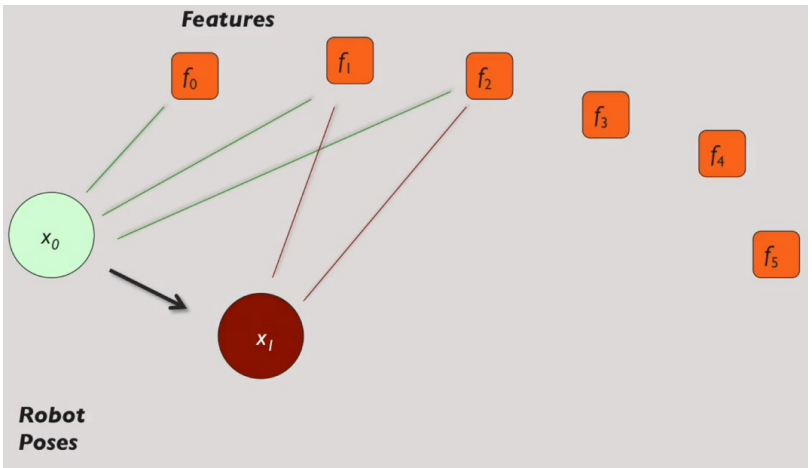
have a vehicle at position $x_0$ and that it's able to, measure features in the environment. $f_0$, $f_1$, $f_2$, etc. at some point in time:



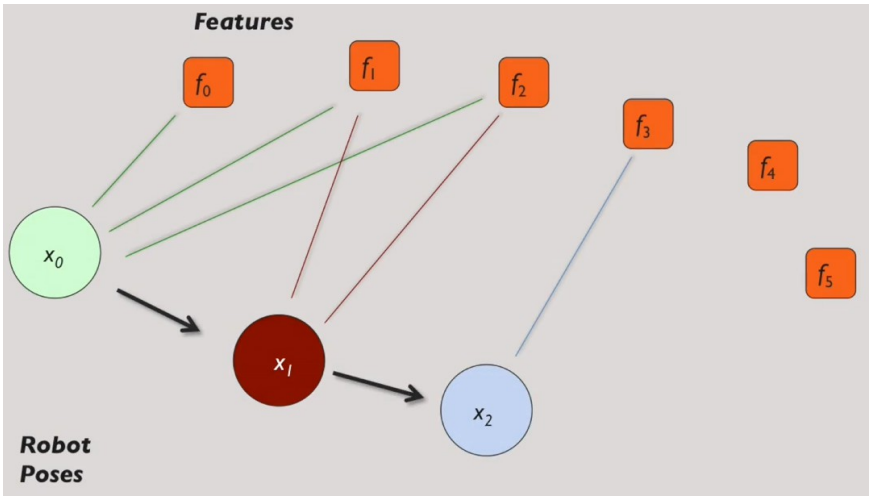

Let's say at position $x_0$ it measures the first three features $f_0$, $f_1$, and $f_2$:



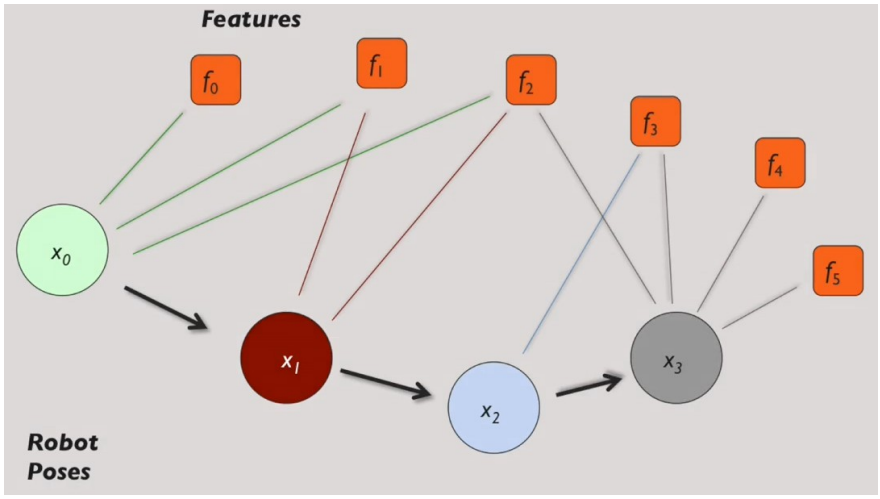

After moving to a new position, $x_1$, it measures a subset of those features, $f_1$ and $f_2$:



The robot moves to another position, and now it measures a new feature, $f_3$, which was not part of its original feature-set:

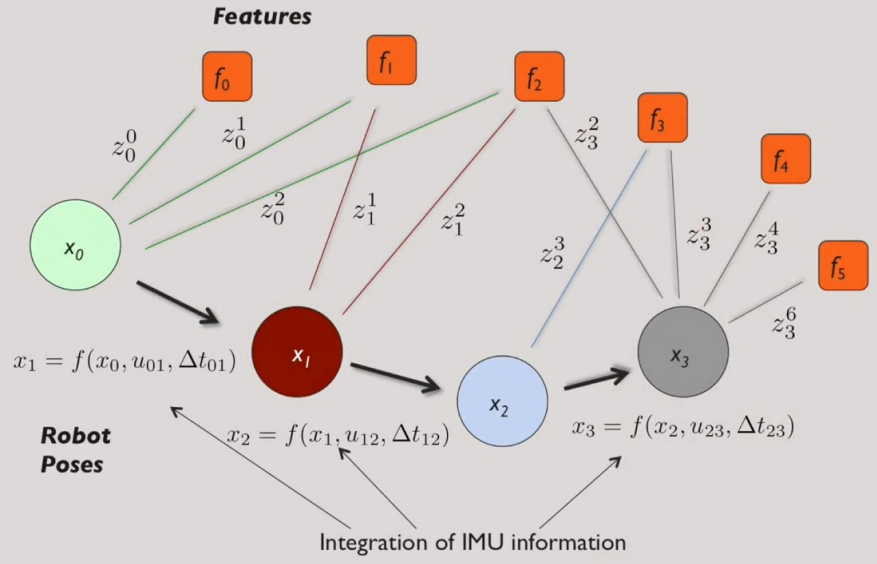



At $x_3$ it now measures four different features, some overlapping with its existing feature-set and some contributing to new features:



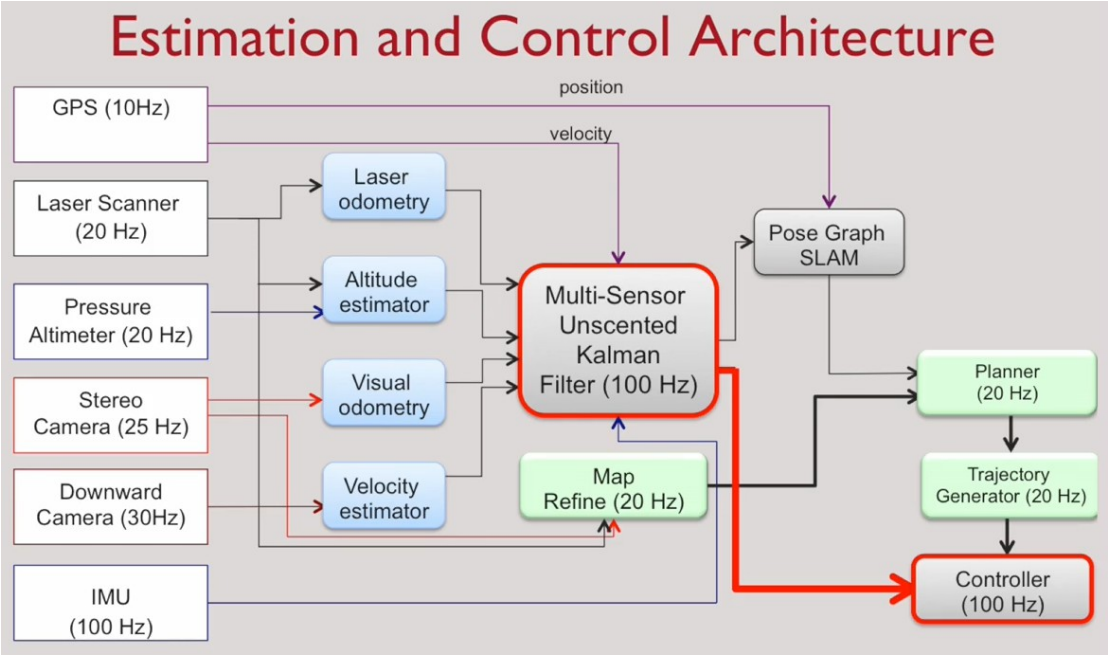

And this goes on and on. The key idea is that we obtain a graph with two different types of edges. The first set of edges corresponds to measurements made by onboard sensors, cameras, or laser scanners. The second set of edges has to do with the movements that the robot has made.

Based on its applied inputs, and based on the time elapsed, it's possible to estimate how far it's moved. In other words, it's able to estimate the difference between $x_1$ and $x_0$, $x_2$ and $x_1$, $x_3$ and $x_2$, and so on. Each of these types of edges corresponds to information obtained from the sensors. This information is noisy, but we eventually obtain a big graph with two types of edges, and the equations that describe these edges.




So we end up with a big optimization problem that we have to solve. If we can solve it, not only can we recover the positions of the features, but also the displacements from $x_0$ to $x_1$, $x_1$ to $x_2$, etc. In other words, the robot can map the features as well as localising itself as it flies through the environment. This is the simultaneous localisation and mapping problem.

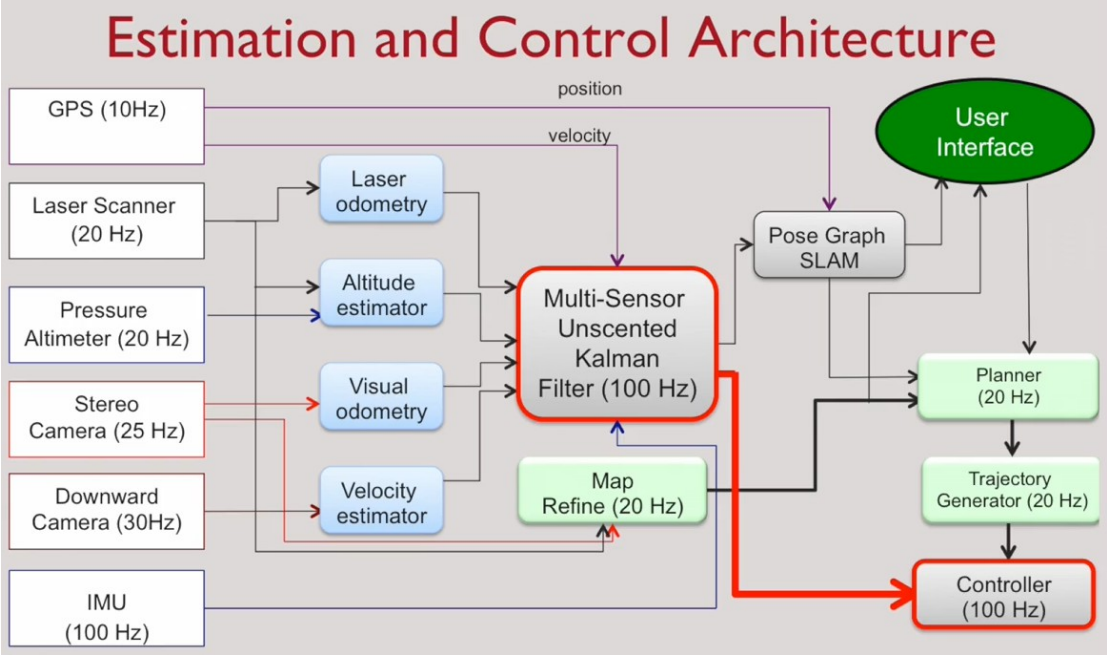
Now putting all of these together involves integrating the information from different types of sensors.




On the left side we have GPS, laser scanners, pressure altimeters, stereo cameras, a downward camera, and an IMU. Each of these sensors gives us information of a different type, and also different rates. The fastest sensors usually are the IMUs, that provide data at 100-200 Hz, while, at the other extreme, the GPS only works at about 10Hz.

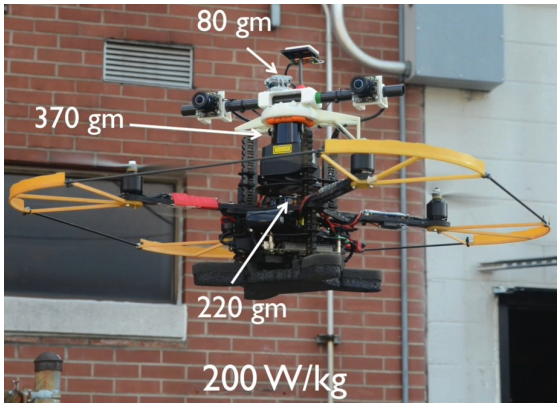
We combine information from all these sensors using a filter. That filter obtains state-estimates at about 100-200Hz, and this allows us to drive the controller that we discussed earlier. In the videos we saw earlier, this state information was obtained by motion-capture camera systems. In this picture, and in what follows, we're going to get the same kind of information at similar rates but from onboard sensors.

In addition to getting the state information, we're able to poll the information from the sensors to create a map and this is the SLAM problem. If we have a user that's interacting with the map, they can specify goals or intermediate waypoints for the quadrotor, and guide the vehicle through a complex environment with a map that's being built by the robot, without actually being in the same location as the robot:




The video demonstrated this basic setup and the algorithms in operation in a set of indoor and outdoor experiments. Robots were seen to build maps with up to 5cm resolution.

A robot operating indoors, or outside under trees or in the shadow of tall buildings, may not be able to use GPS. Inside buildings, it might be too dark for cameras to work the way we expect them to.




We need to remember one basic fact. Robots like this generally burn roughly 200 W/kg of payload. The laser scanner that this robot carries weighs 370g. The stereo camera rig weighs about 80g. The Intel processor & board weigh about 220g. Each of these modules, the processors and the sensors, contribute the total weight of the platform and the total amount of power being consumed. This is one of the drawbacks when we go

to onboard state-estimation. We have to make sure the vehicle has the requisite sensors and the capability to process the information. That drives up the weight of the platform. In this example, the platform weighs 1.75 kg. The larger vehicles that we build are definitely more capable because they have better sensors and better processors. They can also fly longer missions because they carry bigger batteries. However, they lack the advantages that smaller vehicles have. Smaller vehicles can navigate in more complex, constrained, indoor environments, and they're also inherently more agile and manoeuvrable.