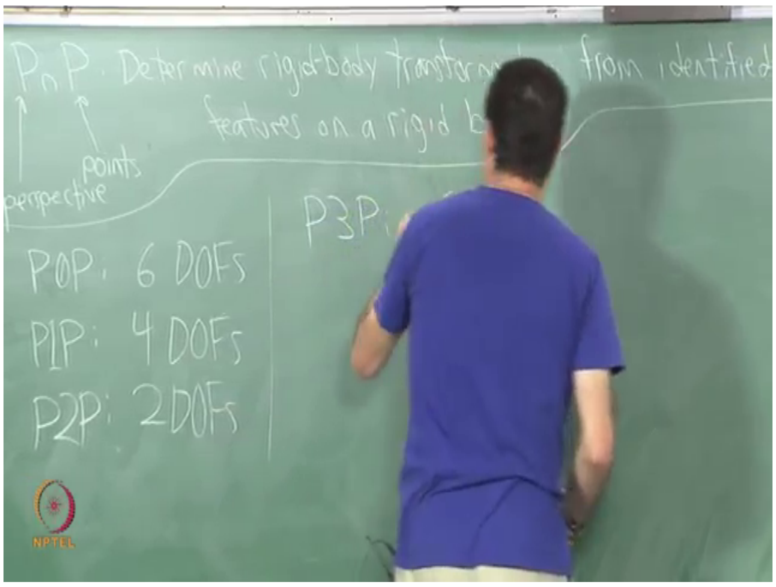
Virtual Reality Engineering
Prof. Steve Lavalle
Department of Multidisciplinary
Indian Institute of Technology, Madras

Lecture – 14
Tracking Systems (perspective n-point problem)

Welcome back, let us continue onward. So, in the last lecture we were talking about tracking systems and I mainly covered the case of orientation only tracking. This is very useful for a fully portable virtual reality headset and where you are completely relying on inertial measurements, inertial measurement units which gives you a gyroscope readings and accelerometer readings and magnetometer readings as well. And as some of you have asked do not you also need to take into account the position of the head, so if you do a motion back and forth like this, or in general if you are tracking other rigid bodies such as your hands then you would like to have position as well as orientation.

So, we talked about different cases and remember we were mentioning that there was the case of line of sight visibility that was the thing we finished on last time. So, we have visibility or line of sight, you know some features that are in view of a camera and then you would like to figure out um.

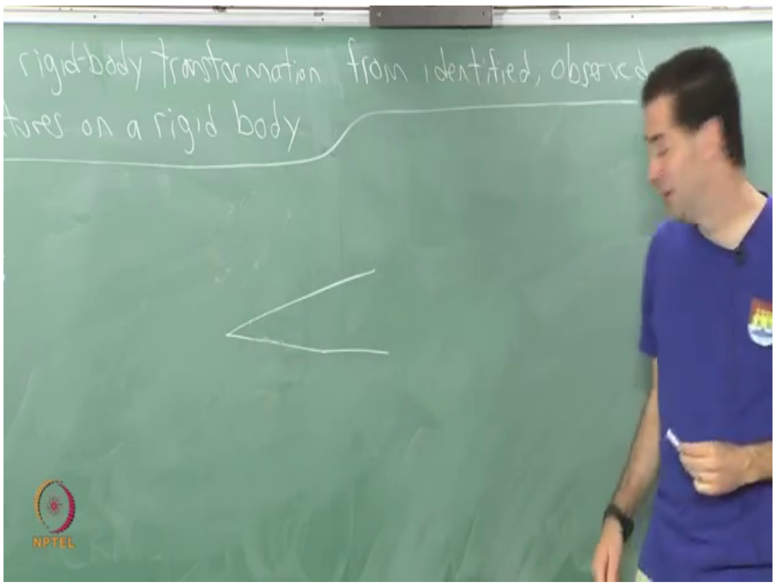
(Refer Slide Time: 01:19)


PnP: Determine rigid-body transform from identified features on a rigid b
points
perspective
P0P: 6 DOFs
P1P: 4 DOFs
P2P: 2 DOFs
P3P:

Where these are in the scene? So, this leads to a very generic and well-studied problem called the PnP problem. Ps the first P is stands for perspective, which is just a model of the camera projection, remember that I said for each feature that you can see in the world. This corresponds to as it strikes through the image plane array that you can narrow down that feature to in the in the world frame.

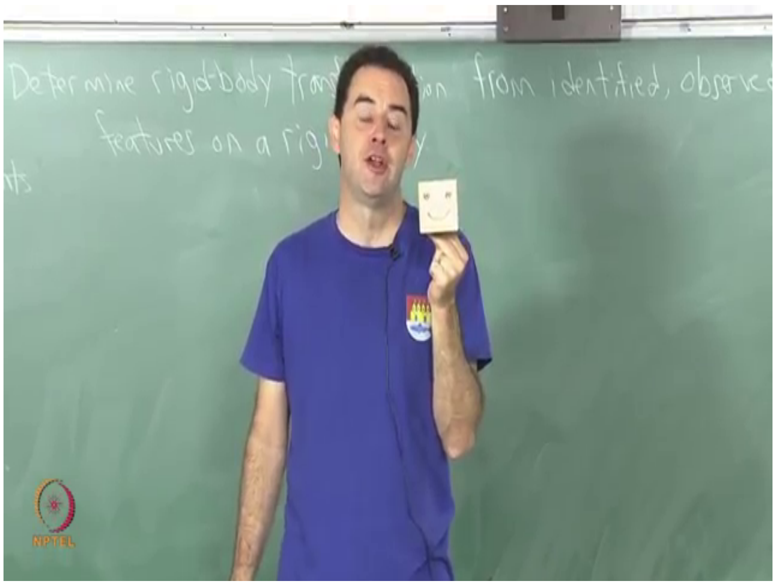
And then the second P is corresponds to points. So, perspective n points problem determine the rigid body transform, the same transform we have been doing all along here from identified, observed features on a rigid body.

(Refer Slide Time: 02:19)



So, in other words I start with the rigid body. So, let me go back to this cube head that I had before.

(Refer Slide Time: 02:52)


Determine rigid-body transformation from identified, observed
features on a rigid body

So, I may have particular features on here and you know perhaps it corresponds each one of my features may correspond to a corner of this, right? I could imagine putting a bright LED on it. And then based on where these LEDs appear in the image, I have the trouble the problem of this figure or this this rigid body has moved somewhere. I want to figure out what it is position and orientation is. What is given to me is exactly each one of the LEDs I know the coordinates of the LEDs in the body frame and I also have labels for them.

This is LED number one maybe on this corner is LED number 2 on this corner and so forth I have labels on all of them. And when I see them in the image I can recover the labels; that is what identified means. So, how can this be done in practice? Well I could have different colored LEDs, right? That would be one way to distinguish them. Another way to do it is to have them flashing in some way and make some kind of code over time so if I see them in multiple frames. They can flash between a light mode in a dark mode. So, they could switch between 2 different frequencies and you can get some kind of coded signal to identify them over several frames.

So, things like that could be done. Let us think about different versions of this problem. And I like to I like to talk about degrees of freedom. So, how many degrees of freedom do we have for the rigid body before we see any features? 6 full 6 right, alright. So, I guess I could say that if we just have P 0 P right n is the number of points. So, if I have 0 point seen then I guess I remain with 6 degrees of freedom for the object. If I observe a single point in the image I just want to reason about the degrees of freedom now.

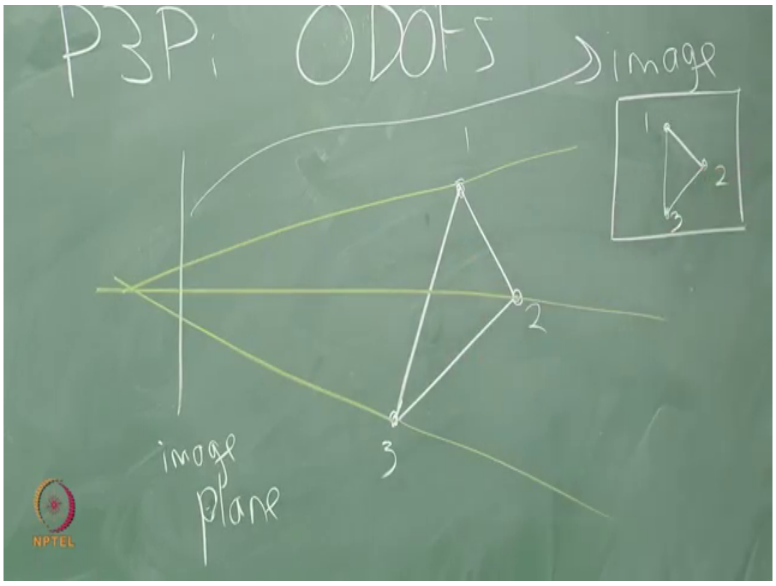
So, for example, I have this object, right? And I can state that one feature is fixed let us suppose it is this corner. So, I have observed this corner I have identified it in the image. So, the question is what can this object do now while keeping this corner fixed in an image? How many degrees of freedom have I lost, have I lost 3? Let us see. So, can I do any rotation of this now, not any rotation because this would be blocked right, but but let us just say in terms of degrees of freedom analysis we just do small perturbation. So, I should be able to do any yaw pitch and roll. So, that is 3 degrees of freedom still intact correct? But what if I make this thing closer or further from the camera moving exactly along the perspective projection line that goes to the point, can I do that too? Yes.

I think I can do that so that leaves 4 degrees of freedom for this. So, what is changed is that this one point did I get it right. This one point can no longer go this way and can no longer go this way because it is been fixed by the pixel in the image. So, so in some sense you can imagine the ij coordinates are the 2 constraints, right? The idea coordinates at this point in the image are the 2 constraints each one of those drops are degree of freedom so that means that there are 4 degrees of freedom left after one point.

So, what about perspective 2-point problem, how many degrees of freedom are going to be left? Who thinks there's a pattern here maybe? So, if I hold 2 points fixed maybe I take these opposing corners here and say that they are fixed I do not know if I can hold onto this very well. So, I guess in that case, it looks like I could spin it like this, right? Is that only one degree of freedom remaining then? It is somewhat difficult to see, but I should be able to also, but do some kind of transformation where I move this back and forth; so that the angle between the 2 rays here right that that stays fixed. So, I am kind of as if these took 2 features are moving along rails, right? And I can go further back while reorienting this and that is another degree of freedom that seem ok. You know try it at home with your own setup.

Maybe I can do it here with a piece of chalk if I just. So, it is like it is like this I should be, I have it like this right I can go like this correct right? So, that is the x degree of freedom try it again here make sure you get that. So, I can go like this, all right? So, we get this extra degree of freedom so; that means, there are 2 dolls left here.

(Refer Slide Time: 07:49)

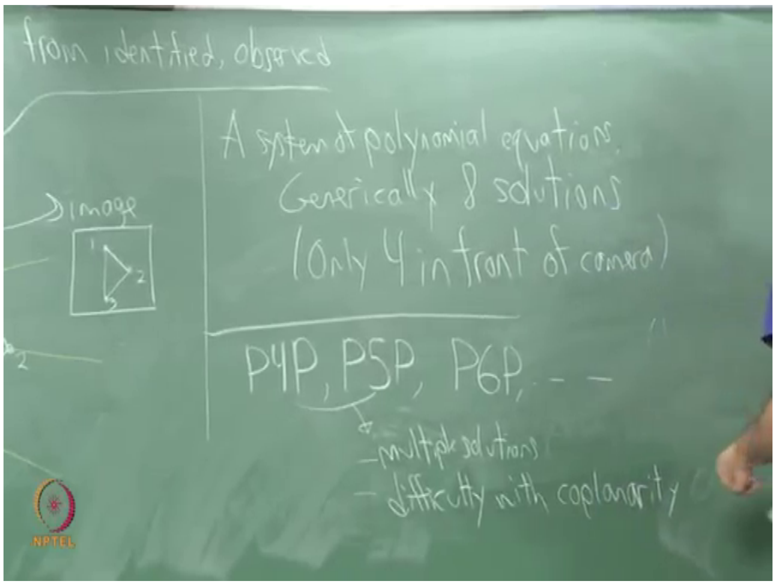


So, they go to P 3 P and as you might imagine were down to 0 degrees of freedom remaining; so, there are no more degrees of freedom remaining and, in that case,.

I have a picture it looks something like this. I have imagined a rigid triangle that has my that has the features on the corners 1 2 3 and then these are all being observed into some image so I will try to make these lines all meet here. So, there is an image plane where these are being observed. So, I get these 3 points observed in the image. So, somewhere in here there is an image all I am really imagining is the detection of 1 2 and 3. So, they are labeled, I know which point is which. And then I can pin this down except it is not the complete picture.

So, in order to figure out exactly where this triangle is located it involves solving some equations. So, you have a system of polynomial equations to solve people have solved this one you can find solutions all over the internet and in books, but the interesting thing that comes up. When you try to solve this to figure out the solution is that you find out that there are generally 8 solutions.

(Refer Slide Time: 09:38)


from identified, observed
image
A system of polynomial equations.
Generically 8 solutions
(Only 4 in front of camera)
P4P, P5P, P6P, - -
- multiple solutions
- difficulty with coplanarity

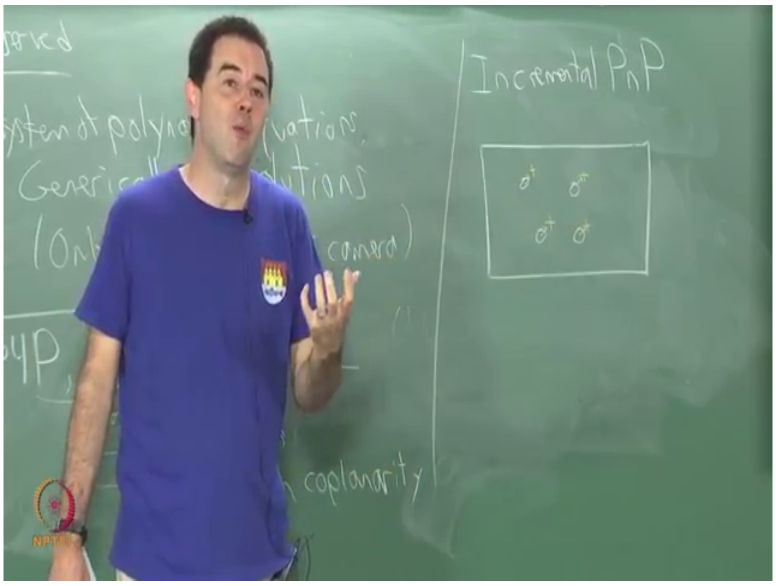
So, you get a system of polynomial equations and there are generically 8 solutions. It turns out that 4 of the solutions will be on the other side of the lines here. So, so you will have 4 behind the focal point and 4 in front of the focal point. So, in reality you get only 4 in front; that is if you set up lines in your in your system of equations here. So, only 4 in front of the camera so; that means, that if I have a fixed rigid triangle and I make a kind of house for it or a cage for it out of these yellow lines then; that means, that there are 4 different ways that I could stick that triangle in there. So, that the vertices of the triangle touch these lines.

So, you can also try that for a home project if you like make you know make a pyramid out of 3 straight sticks and see if you can fit a triangle in there 4 different ways. So,. So, the algebra works out for that, in order to remove redundancy you just go further. So, if we get up to P 4 P, P 5 P there's still some redundancy say, there's still some multiple solutions and problems with top linearity and this special to be true in actual systems where you are observing these points in an image due to discretization and noise. You may end up with several plausible solutions that are very close in terms of the image coordinates, but they might be quite far away in terms of the orientation of the triangle.

So, there ends up also being almost solutions, let us say our epsilon solutions in this range. And so, eventually you get up to P 6 P and higher the more points that you observe the better and the further away from being coplanar the better. So, this provides greater distinguishing power. So, on the oculus rift dk 2 headset for example, they are I believe about 40 LEDs and they are distributed all over the place not necessarily coplanar. So, so this gives a lot of discrimination power in terms of resolving the position and orientation of the headset, when it is in front of the camera. You do not see the LEDs because they are hidden behind infrared transparent plastic.

So, so we cannot see through the plastic, but if you could see in the infrared spectrum then you would see the LEDs. You can also look at it with an IR camera if you want; directly and show the images and you will see the LEDs lighting up, alright just make sense? So, we get enough information we get enough equations to solve, I do not want to go into what happens to what you know how to solve each one of these equations. You start entering into a subject called computational real algebraic geometry. it is so of like the polynomial generalization of linear algebra. So, you know linear equation solving using linear algebra their whole methods for polynomial system solving they they become very useful in ver in various fields and engineering.

(Refer Slide Time: 13:47)



Robot motion planning for example they show up they also show up here in the case of these kinds of solutions. One thing to pay attention to is we call incremental PnP which is suppose I am looking at the image, and in one coordinate frame I have the features.

And I have an estimate of the position and orientation. And then in the next coordinate frame I noticed that these features move by some small amount and I can still you know I still have my identification going. So, these features move by some small amount. All I have to do is slightly uptake lightly update my estimate of the position and orientation, correct right? Because in one frame time let us suppose your cameras running at 60 frames a second so; that means, your heads moving it cannot move too far in 16.67 milliseconds. So, there's a tiny change in the image here and all I need to do is figure out what is the change in position orientation. I have the gyroscope and accelerometer which I can use as well to make a good estimate as far as how much change there has been in the transformation.

And so, to do the final bit of correction I could just perform a very simple local optimization, to just perturb this object I could imagine using the equations of perspective transformation; that I am just perturbing this objects in the virtual world that I am trying to track which in this case is the head. I am trying to track the head and I want to just do a perturbation so that; the the new position orientation that I am estimating lines are perfectly with the observation.

So, I could just proceed in an incremental fashion like. That is like an incremental optimization where I just do a little bit of perturbation in each step and it ends up being fairly straightforward because it is not as hard as completely from scratch, I show you the headset where am I I have to figure out out by solving the equations from scratch. Once I know the solution it does not change by very much from one frame to the next.

So, that is what that part is; questions about that. So, that gives me a new measurement now it is another kind of drift correction information, which is the updated position, so in position estimate that is based on line of sight visibility.

So now, I can use that again in a filtering method, let me just give a little bit of a description of how that works. And then we can move on to another tracking technology called lighthouse. And then I will be done with tracking methods.