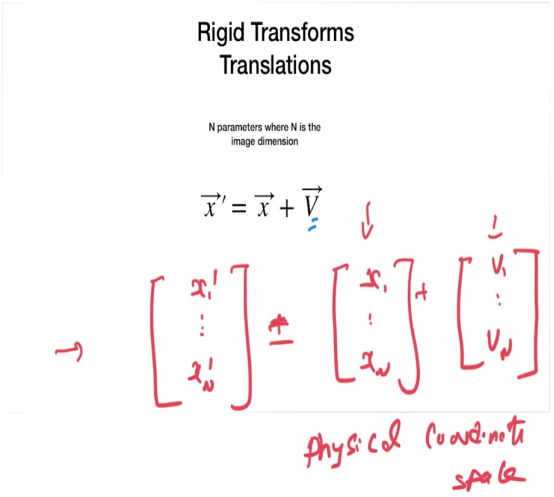
Medical Image Analysis
Professor Ganapathy Krishnamurthi
Department Of Engineering Design
Indian Institute Of Technology, Madras

Lecture 16
Transforms

(Refer Slide Time: 00:15)



Hello and welcome back. So, in this video we will look at the kind of transformations that are typically used in the context of rigid registration techniques. So, the first transform or the most simplest transform and typically a part of all rigid transformation registrations. So, we will look at so called translational transform.

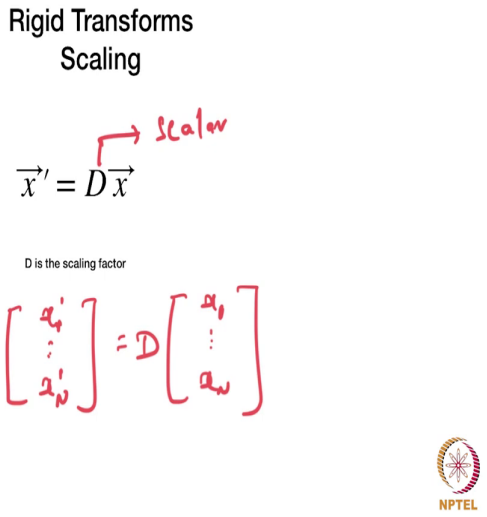
So, we will represent that by this $\vec{v}$, which has to be applied to every point in the physical coordinate space in order to map it to one output. So, I have written it in the form of the vector because area is that, x' and x both have a certain number of dimensions depending on the image.

So, for instance, if a 2-D image x' would be 2 dimensional 3-D image x' would be 3 dimensional. So, if you write it in its more general form, then we will have x_1' up to x_N' for an N dimensional image is equal to ($x_{1.....}$ x_N) + (V_1 ….. V_N). So, this is the most general form in which you can write this, this transformation is typically a very easily managed for, especially for rigid body registration. And it is always the first thing you would try out when you are trying to register two different images.

So, it is a fairly simple enough transform. Once again, remember, all these transformations are applied in the physical coordinates space and not on the pixel space. So, you will have to convert from pixel index to this x, which is your in millimeters. And then of course, this (V_1 V_2……V_N) depend on that for each dimension, there is one translation.

So, you would apply this translation to every pixel in the picture, in order to get the output coordinates. So, this is the most simple transform that you would do in the context of medical image registration.

(Refer Slide Time: 02:34)



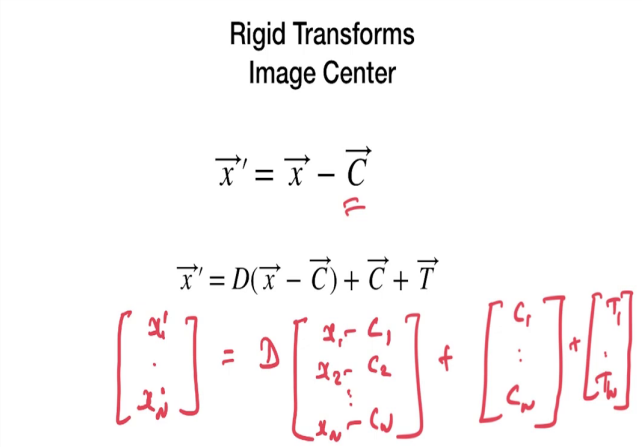
The second transform, which is slightly more complicated, and the first form in the scaling transform. Now, here, there are different versions of it. So, the one we have shown here is where in your applying the same rescaling, this is basically a scalar that you apply to all the dimensions of the image, there is also this anisotropic scaling, where each dimension will get a slightly different scaling factor.

And because much more complicated, this is also difficult to optimize, because this is multiplicative it has nonlinear effects. It also depends on the origin of your coordinates. So, for instance, the origin of the coordinates is in the center of the image, then it will have like a magnification effect. So, but on the other hand, it is basically invites like radiating in all directions. That kind of effect is what you get. But if your origin of coordinates is in the top left, or let us say bottom left.

Then it is scale slightly different affine appearance differs depends on depending on the origin, slightly different effects can be obtained. So, you have to be careful about using this transformation. Typically, it is used when you really do not know the scaling factor involved. Maybe during the image acquisition process, images could be acquired a slightly different dimensions, different magnification.

So, that problem is there. So, the addressed that problem you can use this scaling factor. Once again the most, simplest application is when you are using it using a single scaling factor across all of dimensions. And if you write it in the form of we did for the translation. So, for across every dimension for N is the dimension of the image, you have a unique, you have the same scaling, you do the same. So, you have the accessor scalar.

(Refer Slide Time: 04:24)



So, the other transformation that you typically is overlooked, and I think it is much more important that we take off the annotation thing is the center image center. Now this becomes very important when you are doing rotation. So, typically, what you assume is that you are assuming that you when you do rotations in an image which is again part of a rigid transformation model.

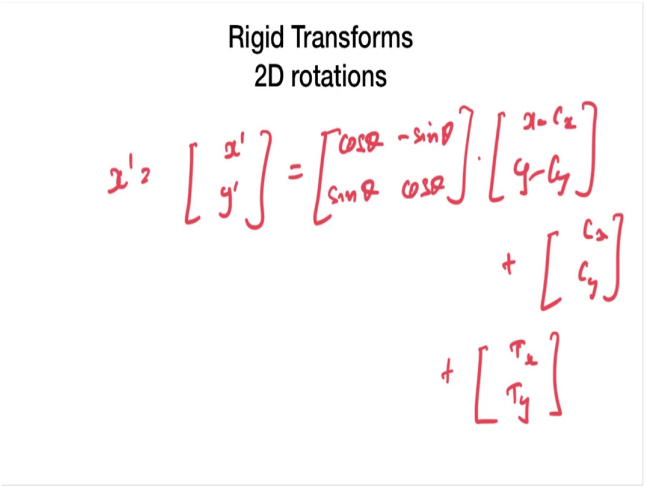
You assume that it is usually done through an axis through the center of the image, but that is not be the case. Sometimes you can also apply rotations with the origin fixed at the bottom left corner of the image. So, that way this annoying the center becomes important.

So, typically, you would not estimate this as part of the registration process, but rather you would just fix the center. So, then before you apply the scaling transform, you would move the center, move the origin to the center, and then do the scaling and then move back before you do the translation.

So, for instance, this particular expression I have written here, it would translate to the following. So, across the dimensions of the image you have a scaling, but you apply after you move plus. And then on top of that, you can also have the translations so, just of course, you have to reset every time you do this.

So, you have to move the origin it fix means that move the origin it means that depending on when you want where you want to, where the object should be when you do the scaling or the rotations you do the following operations. So, there are software packages, which readily implement this. So, even in them, you would ideally fix this is not estimated would fix it and based on and after fixing it, you would do the translation then rotation.

(Refer Slide Time: 06:36)



So, the next thing is the 2-D rotations once again, originally rigid body transformation is not complete without this. So, here the expression would be, for instance, if you are working in 2-D space, typically hold down for a second and shift move it. So, for instance, you would have in this case your x' is basically y', then you have your rotation metrics, you saw this earlier also.

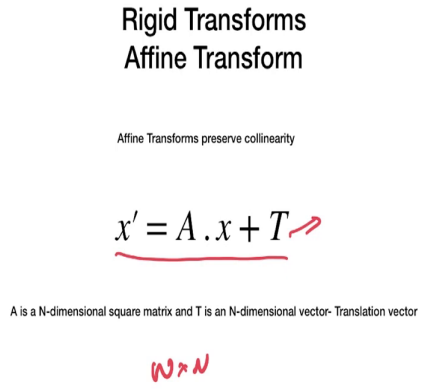
And then you have because once again you would move your coordinates origin of coordinates then of course, add them back after you do the rotation. You also have the option to add translations following all this you can also add after your rotations you can translate. So, this is a typical setup, you do similar things if you want to do this for 3-D.

Once again you have to fix the axis and based on that you would move them fix the axis in the sense, the axis of rotation where the passes through the center of the image or thrown of the corners then you would affect your rotations and then of course, there are any other transformations like for instance in the case of rigid body translation, you typically end up doing the translation.

So, I mean, there are different ways of representing this there saw etcetera, which are which represent these kinds of transformations not go into such details, but we will be understand the parameters involved in all of these are translations here this is minus if you missed that all. So, but this all of this stuff is you must have seen in some of the linear algebra courses or in some other course, we are another medical engineering course fairly straightforward.

But coding this appropriately, applying this correctly is the challenge when it comes to medical images station, but as far as no rigid body registration is concerned or rigid transformation or concern, well studied subject a lot of software's exist the errors are known etcetera. So, you are welcome to try these out in your projects.

(Refer Slide Time: 09:04)



So, the next transformation that we will see is so called affine transform. So, what affine transforms do is to preserve co linearity. So, if there are three points that are collinear in the input space, that is your input image fixed image are three points, which are collinear then after applying this transform still remain collinear.

So, that is the only guarantee behind this transform, this is the general expression for the transformation x' = A.x + T, where N is the N-dimensional square matrix and T is an N-dimensional vector and this is the translational vector one for every dimension of the image.

And so, the parameters are basically N×N coefficients of the A plus d N components of the vector T. See the, affine transform thus both some sharing as a rescue of the image. And of course optimization is always a problem, but typically this is another general transform that you would use in the context of even rigid registration, fed transforms or used.

But of course, the simplest rigid registration model transform is the rotation and translation. So, that is typically the rigid body model that you use first as a first step. And slightly more complicated is this model where you can use, the affine transform correct, you would first start off with the rigid body model, where you only have rotation and translation and then it slowly progress to this fact even scaling becomes the makes the problem more complicated. But like I said before, rigid body registration is well studied.

So, typically, that would be a very good place to start. So, these are the simplest transformations that you can use when you are trying to do rigid registration. And there are, like I said, software packages available which perform this kind of registration and welcome to try those. I will give you a summary of that. When we go to the next week, we also look at non rigid registration. Thank you.