[SQUEAKING]

[RUSTLING]

[CLICKING]

BERTHOLD HORN: OK, let's start. This should be a fun diversion from other things we're doing. We'll develop methods that are useful in photogrammetry and robotics, in particular, dealing with rotation, which has some strange properties.

So for example, in the case of translation, things are commutative. If I go 5 meters in x and 10 meters in y, I end up in the same place as if I go 10 meters in y and 5 meters in x. This is not the case with rotation.

And to illustrate, I suppose that this is the x-axis and this is the y-axis. So I'm going to take this eraser and rotate it 90 degrees about the x-axis like that and then 90 degrees about the y-axis. And I end up there.

And I take the equivalent eraser. And I rotate it 90 degrees about the y-axis and then 90 degrees about the x-axis. And I'm there. So fortunately, that worked. So they're not the same. And so it's not commutative. So it's a little bit more difficult to deal with, particularly when it comes to least squares problems and fitting of data to observations.

So here is an outline of what we're going to be talking about-- properties of rotation, different representation for rotations, and in particular, the one we're going to pick, which is Hamilton's quaternions. And they are more general. But if we restrict attention to unit quaternions, they map directly onto rotations in 3 space.

They allow us to talk about a space of rotation. So for translation, obviously, 3 space is our space of translations. And if we do some optimization, that's the space we're searching in. That's the space we tessellate. That's the space where we get our answers, whereas it's not so obvious with rotation.

Say you want to take an average of force on a seat belt over all possible orientations of a car. That's something that, if it was translation, it'd be trivial. You just integrate over some volume of space, divide by its volume. And you're done. So that's something we'll be able to do.

Applications are, for us, mostly in photogrammetry, although they're very important in robotics and graphics and control of spacecraft. So we'll talk about some of that. And in particular, using this, we are going to develop a closed-form solution to the problem that we started talking about last time, which is either measurements in two different coordinate system of the same object or measurements in one coordinate system of two objects or one object that moved.

So and just for fun at the end, we'll briefly talk about division algebras and space-time. And so here are some of the things I mentioned, where we need to have a good representation for rotation-- obviously, in machine vision, for recognition, and for determining the orientation of an object in space, for the robot arm to, and so on-- so down to protein folding, cryo-- electron microscopy. We all need a good representation for rotation there.

And in particular, with robotic situations, we're dealing with motions in space-- Euclidean emotions, which can be decomposed into translations and rotations. And the Euclidean motions have the property that they preserve distances between points. They preserve angles between lines. And they preserve handedness, which sometimes we forget. But if we don't put that in, then we're going to get reflections.

As you know, most of the biologically important chemicals exist in one particular form. The mirror image form doesn't work. And so it's kind of important to preserve handedness. Put together, those mean that we preserve dot-products. And we preserve triple products.

And we're leaving out, by doing this, transformations of free space that would otherwise be perfectly reasonable, such as reflections and skewing and scaling. We'll talk a little bit about scaling. But mostly, for the moment, we'll leave it out.

So rotations, first Euler's theorem-- any rotation of a rigid object has the property that there is a line that is not changed. That's the axis. Then there's the parallel axis theorem, which says that any rotation about any axis is equivalent to a rotation about an axis through the origin plus a translation.

So for us, it's going to be very convenient to separate translation and rotation this way that when we talk about rotation, that we're trying to recover-- we're just going to talk about rotation about an axis through the origin, because we deal with the translation separately. And because of the parallel axis theorem, we can do that.

Then if we figure out how a sphere rotates, we've basically figured out how everything rotates, because we can-- and that sort of simplifies thing. We can just think of a sphere rotating in space and all the possible ways that can happen. And that, then, corresponds to rotation of space.

When we talk about attitude, we basically mean orientation. We mean orientation relative to some standard. So for example, we might have models as we had when we're talking about the patents, except now in 3D. And they'll be in some preferred coordinate system.

And when we talk about locating that object, we're talking about finding its position in space and finding its rotation relative to that reference rotation. So by the way, this is going to be on Stellar, this whole presentation. Then there's a four-page blurb that summarizes everything you want to know about quaternions. Or maybe not. Maybe everything you need for this course.

Degrees of freedom. So we talked about this last time, that there happened to be 3 degrees of freedom to rotation, and that can be slightly confusing. What's interesting is that rotational velocity is much easier. Angular velocity-- all we need is a vector that gives us the axis and a rate, so many radians per second.

And if we take the axis and multiply it by the rate, we have a 3 vector. And those 3 vectors add, just like translations. So if you're spinning around the x-axis at 1 radian per second, you're spinning about the y-axis at 1 radian per second, then you're spinning about the 1, 1, 0 axis. So it's very easy to just add these up.

And of course, that can change. So now, you've moved a little bit. And the next time instance, the axis will be different, So but instantaneously, rotational velocity is very easy. And this, for those of you who know about these things, corresponds to the Lie algebra versus the Lie group. The rotational velocities or the Lie algebra. It has all the nice properties of an algebra. The rotations, which are sort of the integral of that, don't. They don't have quite those nice properties.

Poisson's formula tells you what the velocity is of some point when there's a known rotational velocity-- right-- rotational velocity. So omega is that vector, the axis of which is the axis of-- the direction of which is the axis of rotation. And the magnitude of it is the rate of rotation. And then if we have some point, r, we just take that cross product to figure out what the velocity is. And obviously, it's going to be perpendicular to r. And it's going to be perpendicular to omega. And you can get its direction by using the right-hand rule.

So rotational velocity is simpler. They add. And we just saw finite. Finite rotations don't commute. And we talked last time about how they're n times n minus 1 over 2 degrees of freedom. And for n equals 3, that happens to give you 3.

So often we talk about rotation about axes like I just did-- rotation about the x-axis, rotation about the y-axis. And if you follow that view, then you might think that in n dimensions, rotation is n dimensional. And therefore, it is better to think about rotation as rotation preserving certain planes.

So for example, we can preserve the xy plane. We can rotate in such a way that the xy plane is not changed. Things in the xy plane are moved into xy plane somewhere else-- yz plane and zx plane-- also, 3. But that one works, because in 2D, we only have the xy plane, 1 degree of freedom. And in 4D, we have six combinations. And that's the number of degrees of freedom of rotation in 4D.

So before we go further, here's an interesting thing that we will exploit. And also, it's a kind of preview of a method we'll use, which is cross products. Cross products, as I mentioned, are kind of confusing, because you take the product of two vectors. And you get another vector, which isn't quite right. But it happens to have 3 degrees of freedom. So we represent it as a vector.

And again, that's a coincidence. What is it? Well, it's perpendicular to the two vectors you started with. And so in higher dimensional space, if you take-- if you say, what are the things that are perpendicular to two vectors, well, they're not going to be a vector. There's going to be a whole subspace. So the generalization of a cross product would be something more complicated.

Like, one way to generalize it in an n-dimensional space, is it's that vector that's perpendicular to n minus 1 other vectors. Well, here, n minus 1 happens to be 2. So that's the way we get our cross product. And it's actually very convenient for certain operations, particularly, matrix vector operations, to represent the cross product in this way, where we take the skew-symmetric matrix. And we can represent A cross B by multiplying that skew-symmetric matrix by B.

And it has the right degrees of freedom, right, because A has 3 degrees of freedom. A skew-symmetric matrix has 0 on the diagonal. And the off-diagonal is symmetric. So you again have 3 degrees of freedom. So that seems like a reasonable isomorphism.

And then once you've done that, you've gotten rid of this weird thing, the cross product. And you have something that you can manipulate in combination with other vectors and matrices. Now, of course, why did we pick A to expand as the matrix? We could equally well have picked B and have written it this way, equally valid. And you'll notice this matrix is a transpose or negated version of that matrix. So the matrix for B-- so it's order dependent. And we know that, because cross products don't commute.

So there are loads of representations for rotation, which tells you that there's a problem. With translation, we just have xyz, be done with it. So one notation that's very useful is axis and angle. So we have an axis, which can be given as a unit vector. And we have a number of degrees that we rotate or radians that we rotate through. So that's 3 degrees of freedom, because the unit vector only has 2, and plus the angle gives us 3.

Now, sometimes we can derive other notations from that. One of them is the Gibbs vector, which tries to combine those two into a single thing. So this is sort of inconvenient in a way, because you've got a 3 vector, which happens to be a unit vector. And then you have this thing.

So the Gibbs vector is a vector-- and it turns out that you don't want to just multiply by theta. That doesn't give you a nice algebraic properties. It's much better-- plus theta wraps around 2 pi. So that doesn't really make much sense anyway.

And Gibbs found out that you can get some useful properties by taking tan theta over 2. So now, you have a vector. It's no longer a unit vector. So it has three components. So it has the right number of degrees of freedom. Only bad thing is it blows up at theta equals pi, right, because 10 of pi over 2 is infinite. And theta over pi-- so is that a problem?

Well, rotating through 180 degrees, that's a perfectly reasonable thing to do. So it's not like we want to exclude that, because it's not a reasonable thing to do. So that is a problem. And it's very similar to y equals mx plus c when the line happens to be perpendicular up, parallel to the y-axis.

Then we have Euler angles. And there's a classic text in mechanical engineering by Gold which explains about Euler angles. Basically, you rotate about x. You rotate about y. You rotate about z. If you're in an airplane, you have roll, which is rotation about the axis-- length-long axis of the plane. Then you have pitch, which is going up or down. And then you have yaw, which is going left and right. So it's very convenient to think about that.

But as you know, the result will depend on the order. So yeah, if you have very small angles of roll, pitch, and yaw, which is a comfortable airplane flight, it probably doesn't matter a lot, because you get pretty much the same result if you compose them in different orders.

But if you have large angles, like 90 degrees, it's definitely not going to work. So that means if you define Euler angles, you've got to do several things. One of them is you have to define the order of operations. Does the x-axis come first and so on-- plus other rotations about the new axes after you rotate or about the original axes.

Anyway, as a result of this, as Gould points out, there 24 different definitions for Euler angle. And there hasn't been an international meeting to say, thou shalt use this one. So it's pretty confusing. Then we have, of course, orthonormal matrices, which I'm sure you're all very familiar with. And they have these two constraints on them.

And then this one is a kind of exotic one. This represents a rotation matrix in an exponential form. So what does this mean? Well, here this capital omega is that 3 by 3 skew-symmetric matrix that you get for cross product.

And so what does it mean to have exponential with a matrix in the exponent? Well, you just use the formula for e to the x, except now, x is the matrix instead of scalar. So you're going to end up with 1 plus x plus 1/2 x squared and so on, where x now is a matrix. So you can make this work. But we won't pursue this.

And the others-- so here's another one. Stereography-- so there are lots of projections of the sphere. And there's a whole cottage industry dating back hundreds of years, because people found it necessary to take the spherical or almost spherical Earth and map it onto a plane. So they could sell maps. And so there are lots of these.

One of them is conformal-- that is, it preserves angles. And it's called stereography. And as we said, a rotation of a sphere induces rotation of the space. So we can take a sphere, map it onto the plane, then mush things around in the plane, and map them back onto the sphere.

And if we mush things around just the right way in the plane, then we actually have induced the rotation on the sphere. We'll look at a picture of that. Again, we're not going to use that. But it's kind of amusing. The thing you do in the plane is to treat the plane as a complex plane and do some homogeneous transformation on that.

And then, of course, physicists have to invent their own stuff. So there's a type of 2 by 2 complex matrix that can be used to represent rotation that Pauli came up with, the same man who's famous for saying, that paper isn't even wrong as a criticism of one of his colleagues.

So 2 by 2 complex numbers-- that sounds like 8 degrees of freedom. Well, they're not just any. They have to be Hermitian. And they have to be unitary. So it turns out that after you do that, there are only 3 degrees of freedom left.

Then we get to Euler parameters, sometimes also called Gonzalez parameters-- Rodrigues parameters. Excuse me. And then we get to unit quaternions. Now, of course, there's a relation between all of them. And in fact, it's important to be able to convert between them. And we'll do some of that.

And it's a little bit like many other problems, where you're trying to solve something. And it turns out, if you have the right representation, the answer is easy. And so often, the problem, then, ends up being the conversion.

And so then you find that, well, the conversion is-- just like in the trivial case, we could have polar coordinates versus Cartesian coordinates. And maybe the answer in polar coordinates is trivial, but it's not in Cartesian coordinates. And then you're left with a conversion. Well, in that case, the conversion is not too hard. But here, sometimes it is.

So let's start with that. This is Rodrigues. Rodrigues was a banker in Paris. And apparently, he found banking a fairly boring profession, because he'd spend his nights doing math. And so this is one of his results.

Basically, the operations we're going to want to do are two. One operation is take a vector and rotate it, or find out what that vector is in a rotated coordinate system equivalent problem. The other one is composing rotations. We've rotated a little bit in this-- around this axis. Now, we're going to rotate about another axis. What is the overall result? And that turns out to be non-trivial. So this, here, is addressing that first problem.

So we have a vector, r. And we have an axis of rotation that's here vertical. And we're going to rotate through an angle theta about that axis. And that will take us to a new position, r prime. And the question is, how do you compute r prime?

Well, if we first convert it to a 2D problem in this plane indicated by this ellipse, then it's very easy, because we just take this vector and that vector and combine them in a weighted fashion. So this dotted vector is this vector times cosine theta plus this vector times sine theta.

So you can see how, when theta is 0, the dotted vector will be equal to this one. And when theta is equal to 1-- pi over 2, it'll be equal to this vector. And all the way along, it retains its length and so on. So in 2D, the rotation as we know it is very simple. And like mathematicians want to often do, we reduce a complicated problem to a simpler one we already know how to solve.

But that's in this plane. And so we first have to figure out what these two vectors are that we are combining using cosine and sine. And so first of all, this one here-- so how do we get that one? Well, we take r. And we subtract out this vector, right? And the difference is going to be this vector here.

Well, that means that we need to know what this vector is. Well, that's just projecting r onto omega. And since omega is the unit vector, it's very easy-- there is a formula. And so therefore, this vector is r minus that thing.

And then finally, we need to know what this vector is. And basically, that vector has to be perpendicular to omega. And it has to be perpendicular to this, because in this plane, it's pi over 2 away from that vector, right? So by taking the cross product of-- let's see-- this thing here with this thing here, we end up with, after simplification, just omega cross r. And it makes sense, because it's going to be perpendicular to r. And it's going to be perpendicular to omega.

Anyway, you put all of those things together. You get this formula. So r prime, the vector after rotations, is cosine theta r, the vector before rotation, plus these terms. And you can see that rotation is not quite that trivial.

So anyway-- so axis and angle is a useful notation. And it's often a good way to visualize things. It's easy to say, OK, we're rotating about the x-axis through 90 degrees-- easy to understand what it is. What's a disadvantage? The disadvantage is composition. If you have two rotations given as axis and angle, how do you combine them into a single?

We know by Euler's theorem, there's a single rotation that represents the combination. But how do you combine them? Well, it turns out what I would do is convert both of these to some other form, like orthonormal matrices. Multiply the orthonormal matrices. Go back to this form. So that's a disadvantage. There is no-- you can rotate vectors. That's the one operation we want. It's hard to do composition of rotation.

So OK, well, we'll skip this. This is the exponential-- the algebra view of things. And that, too-- let's forget that. I just wanted to point this one, which we mentioned. So here, the idea is that here's our sphere that we're going to rotate. And we project it onto a plane, which happens to be tangent at the North Pole.

And this projection-- if your center of projection is the South Pole-- so you go from the South Pole to a point on the surface and that-- then extend that line until it hits that plane. And you will see a couple of things. One of them is you can map the whole sphere onto a plane-- well, except the South Pole, right?

The other one you can see is areas are going to be distorted, right, because if we're dealing with, say, Alaska up here, it's going to be mapped onto the plane without much magnification. But if we're looking at Australia, it's going to come out way over here. It's going to be huge. So this particular map projection is actually used for the polar regions, because most of the other map projections don't work very well there. But it does have that feature.

And why is it liked so much? Well, because it doesn't distort shapes. So it preserves angles. Most-- all other projections have the property that they will distort shapes. They will distort areas. You have your choice. You can pick projections that preserve area. You can pick projections that preserve shape angles. But you can't do both.

OK, so we start off-- we take our sphere. We projected it onto this map-- this plane, which is the complex plane. Then in that complex plane, we do this operation. So z is a complex variable coordinate in this plane. And z prime is its new position.

So and a, b, c, d are complex numbers. So everything in that plane is going to get moved around. And now, we project it back on the sphere. And the amazing thing is that the pattern on the sphere is just the rotated version of what it was before. And therefore, we can think of rotation in 3 space, instead, in terms of some operation in the complex plane, which is pretty amazing. So that's yet another representation.

OK, so what are the desirable properties? We already mentioned some of them. We need to be able to rotate vectors or equivalently rotate coordinate systems. But also, we need to be able to compose rotations. We would like to have an intuitive representation. Like, axis and angle is sort of intuitive. It's easy to understand what that is, whereas if you look at the rotation matrix, you look at the numbers-- what do they mean? I don't know. You multiply a vector and see what happens. But the numbers don't directly mean anything.

If you know a little bit more about it, you say, well, if I look at the trace with the sum of diagonal elements, it's-- I don't know-- 2 times 1 minus the cosine of the angle of rotation. But it's not very intuitive. Then is it redundant? Well, orthonormal matrices are a prime example of something that's definitely not redundant, because we have 9 numbers to represent three things. Yeah.

Right, that's a very good point. So so much for my story about it not being intuitive-- well, if you think about rotating the x-axis, i.e. 1, 0, 0-- multiply the matrix by that-- you get the first column. So as he's pointing out, the first column is a significant vector. It's what happens to the x-axis. Second column is what happens to the y-axis. Third column is what happens to the z-axis. So you can understand the rotation matrix that way.

And similarly, you can talk about the rows going in the other direction. But you wouldn't know the axis of rotation or the angle of rotation. So but yes, that's a good point. OK, redundant-- so let's see. Axis and angle-- well, axis and angle is kind of redundant, because if we have a vector, it has three components plus an angle is four.

Gibbs vector is not redundant, because we've multiplied the vector by the tangent of theta over 2. But we don't want singularities. Gibbs vector has a singularity. So we'd also like it to be computationally efficient. And we'll get into that a little bit. And we want to be able to do lots of things, for example, interpolate orientation.

So in graphics, we might often have some person performing some dance or action in the world. And we would like the artist to only have to specify certain distinct points rather than every frame. And so we'll need to interpolate.

Well, if that person is rotating, how do we interpolate? How do you do a partial rotation? And so you might say, well, half a rotation-- that's the square root of the rotation matrix, right, because if you multiply that by itself, you get the rotation matrix. But how do you get the square root of a matrix? So that's something.

Another thing we might want to do is get averages over ranges of rotation. For us, a big deal is going to be optimization. We're going to try and fit coordinate system. We're going to try and fit measurements to models. And in that case, we need to be able to do things like take a derivative and set the result equal to 0.

Well, the derivative with respect to the rotation matrix-- that doesn't make a lot of sense. What is it? Well, something with nine components. But how do you enforce the constraints? Then in some cases where we can't get a closed form solution, we might want to sample that space and try some subset. But we'd like to sample the space in an efficient way, which means we want to sample it uniformly.

So if there was a translation space, we'd just chop up space into voxels equally spaced. And we'd be done. Or if we go into-- for a random sampling, we could just generate a random x between the range and the random y between-- in a certain range and the z in the certain range. But how do you do that for rotations?

Now, if you did it in Euler angles, you could say, OK, well, I'll take the first angle between minus 90 and plus 90 and then the second angle and so on. And you get into non-uniform sampling of the space. For example, a much easier example is on the sphere, we can use latitude and longitude as coordinates. But if you use that to control your sampling, you're going to sample much more closely near the two poles, right? So that's one-- add one more dimension, and then you get the problem for rotations.

So the idea of having a space of rotations is very attractive, because then we can tessellate that space. We can sample it uniformly or randomly. We can compute averages or whatever. So here are some of the problems we already talked about. The orthonormal matrices-- they're redundant. And the constraints are complicated. Euler angles don't make it easy to compose rotations, just as with angle and axis notation. Best thing you can do is convert it to orthonormal matrix and multiply.

And then there's gimbal lock. So I know you're all too young for this. But there was a time when man went to the moon, which was a very exciting time. And when you listened on television and you ignored the stupid commentators that were talking over the communication from the cabin, you'd hear them talk about approaching gimbal lock. So what was that?

Well, they-- Draper Lab built their navigation system. And it had gyroscopes in it. And gyroscopes basically have three-- that type of gyroscope has three axes. And it has a part that is spinning at high speed and wants to continue spinning around that axis. And then the cage basically allows the spacecraft to rotate about that fixed direction.

But as with Euler angles, you can get to a situation, where two of the axes line up. And then you've lost a degree of freedom. And if you move through that point, you've basically lost your orientation in space. So they were carefully instructed not to go through that point, because their automated systems wouldn't work correctly. And so that's exactly the problem with Euler angles, that when you rotate one of them through 90 degrees, it lines up with the next one. And you've lost a degree of freedom.

And so by the way, how do you solve it? Well, the easiest way is to have four axes. And in that case, unfortunately, you can no longer treat it as a passive system. You have to have an active system that drives the axes so that you never have two of them line up.

Anyway, Euler angles-- Gibbs vector had that singularity. Axis and angle has the problem of composing rotations. And then of course, we don't have a clear idea of what is the space of rotations. So we get to Hamilton, who lived in Dublin, Ireland. And he was fascinated by algebraic couples.

So the view is that we think of complex numbers as pairs of reals. And can we somehow generalize that? And you want to have a nice algebra. You want to be able to add, subtract, multiply, and very important, you want to be able to divide. I mean, you can do all sorts of wonderful things. But if you can do that, then you can solve equations by subtracting, cross multiplying-- doing all those things we're used to.

And so there was a lot of interest in physics, particularly in dealing with vectors in space. Well, they weren't called vectors, but dealing with points in space. And so the natural thing was, let's go from real numbers to complex numbers to triples.

And then what can we do with the triples? Well, you can multiply them. And of course, the dot products not much used for this purpose, because it's a scalar cross product. Well, the trouble is if you've got a cross b, c equals a cross b. Can you say a equals c divided by b? No. So that was his problem.

And it really puzzled him for quite a while. And he trained his children to, every morning he came down for breakfast, to ask him, papa, can you multiply triplets yet, which were these vectors-- we later call vectors. And then on this fateful day, he got it. And he went with his wife on his usual Sunday stroll through Dublin. And he committed a criminal act on the bridge. He engraved in graffiti the basic equation you need to solve this problem without explanation, just the formula.

And then later when he described it, he's saying, "And here, there dawned on me the notion that we must admit, in some sense, a fourth dimension of space for the purpose of calculating with triples. An electric circuit seemed to close, and a spark flash forth."

Remember, it was 1843. So there's a lot of excitement about electric stuff but long before Tesla and Edison. And the key thing was that he decided you can't do it with three. There's no way to do it with three. But you can do it with four.

And so then he wrote a book, which is basically incomprehensible. It's like 800 pages of heavy math. But we'll simplify it a lot. So the insight was, you can't do it with three components. And then he took his cue from complex numbers, where we introduce a, quote, "imaginary" number. And he said, well, maybe if we have more of them, maybe there are other things, when you square them, they become minus 1. And so that was his key insight. He used the notation ijk, such that i squared and j squared and k squared are minus 1. And then very importantly, he added this ijk equals minus 1.

And this is what he engraved in the bridge. And it's still there, although heavily worn by mathematicians touching it to see if they can get the inside from-- anyway, from those you can get everything else. For example, if you want to know what ij is, well, you just multiply ijk by k. And you get k squared, which is minus 1. So it's going to be minus 1 times-- and so on. So you get k So these others are all subsidiary. He didn't bother engraving them.

He just took the-- importantly, multiplication is not commutative. So ij is k ji is minus k. And that sort of corresponds to cross products. So this kind of suggests a connection with vectors. And so there are lots of different ways of thinking about quaternions.

One is just simply as a real part and three different flavors of imaginary parts, rather than just one imaginary part. But then you can take these three and think of them as a vector in space and treat the whole thing as a scalar, q0, plus this vector and use this notation , which has a scalar, comma, vector part. Or you can just write it as a four-vector, a thing with four components.

So I use this notation with a little circle over it to denote a quaternion. It's not standard. So don't get confused by that. But when I'm writing stuff, it's very important for me to distinguish the scalar, the vector, and the quaternion. So I use the little circle to denote that.

Another way of thinking about quaternions is as 4 by 4 matrices. And that brings us back to that isomorphism we had that was useful for cross products. There would be an isomorphism of quaternions with 4 by 4 matrices that allows us to do multiplication.

And yet another way is to think of a complex composite of two complex numbers. So complex numbers are real plus i times the real. Now, imagine that you replace the real with complex numbers. You've got something with four components, because you've got to be careful, because you've got two different types of imaginary things. That's actually a way we aren't going to use. But it's a way that leads you to other things.

For example, you can use that idea again. And now, you get, instead of four components, eight. And then again, you can get 16 components. So you can build these algebras that have 1, 2, 4, 8, 16 components. And they get weirder and weirder. We're going to stick with four. That's as far as we need to go.

OK, so here's the view of multiplication using Hamilton's basic explanation of how to multiply these things. Here are two quaternions. You're going to multiply them. And obviously, we're going to get 16 terms. And then j times k is minus i. And so we can gather them up as, again, a scalar part and then these three imaginary parts.

And that's the bottom baseline truth. But it's kind of too detailed. It's kind of hard to keep-- remember that. And it's, for us, who now-- remember this was before vectors. But today we're all into vectors and scalars. So in that notation, it's a lot easier to write it this way. So here's a quaternion p, a quaternion q. We multiply them together. And we get this quaternion. And the scalar part of it is just the scalar parts multiplied minus the dot product of the vector parts and so on.

And it's non-commutative. Why? Because it includes this cross product at the end here. So if we interchange these two, we're going to get q cross p, which is, of course, minus p cross. OK, so this is actually a way we're going to use it. Just it's much more compact than this.

Here is another notation that sometimes very useful. And this is analogous to the isomorphism we had between vectors and the skew-symmetric 3 by 3 matrices. Here, we have a isomorphism between a quaternion and a 4 by 4 matrix, which happens to be orthogonal. And if the quaternion is a unit quaternion, it's also a unit normal. So this, actually, in our case is typically a orthonormal 4 by 4 matrix.

And so this multiplication up here can be written as this product of a matrix times a vector. And there's quite a bit of structure to this. You can see the first column is just the quaternion p. And then you can see that it's sort of skew-symmetric, except for the diagonal, which would be 0 if it was skew-symmetric.

And so we can write the product of two quaternions in that form. So we can write pq-- so this is just like cross product, where we said a cross b is some matrix times b. Similarly here, pq can be written as this matrix times q. And the matrix looks like this. And the matrix is orthogonal. And it's normal if it's a unit quaternion. And if we get rid of p0, then it's skew-symmetric.

And again, just as with cross products, we have the choice of either expanding the first part into a matrix or the second part. So here's the other version, where now, we've turn q into a 4 by 4 matrix. And this 4 by 4 matrix looks a lot like this one. The first column is pretty much the same. The first row is pretty much the same.

What's different is this 3 by 3 submatrix is flipped. Its transpose. So they're very similar. But and this corresponds, again, to the fact that multiplication is non-commutative. If those two pieces were the same, then this whole operation would have been commutative.

So now that we've got the basics and different ways of representing it, it's easy to prove some of these basic results. So first of all, it's clear that it's not commutative. And that's why we're talking about it. We wouldn't be talking about it if it was a commutative because then there is no way that it can represent rotation.

It's associative, and that's not too hard to prove. Just use the formula for multiplication. It's just a bit of messy algebra. Then we define a conjugate. And the conjugate in analogy with complex numbers just means you negate the imaginary part.

And then the next result you can get is that the conjugate of a product is the product of the conjugates in reverse order, which is just like matrix multiplication-- AB all transposed is B transpose A transpose. So there are some analogies here. And if you into physics, you realize that different heroes of physics had different ways of approaching quantum physics. And some of them like to do it with the complex number view. And some of them like to do it with a matrix view. And you can find that they're kind of equivalent.

Yeah, so I've done a little of that. And I'm going to do more of it. And the main reason is that we can get a closed-form solution, because we can differentiate with respect to a quaternion. We can't differentiate with respect to orthonormal matrix. So that's kind of the main reason.

But there are these subsidiary reasons, like if you can't get a closed-form solution, you can do a search in the space of rotations. And you can sample that space efficiently, either uniformly or randomly once you've got the notion of a space of rotation, which is the case with quaternions.

OK, so a few other dot product-- well, the dot product, if we think of it as a four-vector, it's just the ordinary dot product of a four-vector. If we think of it in terms of scalar and vector, well, then it's the product of the scalars plus the dot product of the vectors. But underneath, it's just q0 squared plus qx squared plus-- sorry-- p0 q0 plus px qx, et cetera.

And therefore, there's a norm. We can define a norm. And this is perhaps the most important. If we multiply q by its conjugate, it turns out, we get a real quantity. There's no imaginary part. And it's this. It's the dot product of q with itself times e. Well, e is this quaternion that has a scalar-- it's a scalar 1. And I could have just written 1 there. But that looks sort of funny. So I invented this symbol, which is a quaternion that has no vector part.

And why is this important? Well, if you can do that, then you have division. So the multiplicative inverse is this. And that was the problem with the triplets. There was no way of defining an inverse. And here, we can define an inverse, well, except for q equals 0. But then that's always a problem.

So and so this means, for example, that we can get the inverse of a rotation very easily. First of all, in the rotation, we're going to be dealing with unit quaternions. So q dot q is 1. So in the case of that unit quaternion representation, the inverse is just the conjugate, sort of like in the case of orthonormal matrices. The inverse is just the transpose.

OK, and then there are a few other sort of minor properties. And you can just take those on faith. Or you can check them by doing the multiplications. So the dot product of products is the product of dot products. All surprising that. And then this first line is the special case of the second line, which is more useful in calculations.

And then this one's pretty handy. So what's happened here? Well, sometimes we have a quaternion on one side. We'd like to move it to the other side of a dot product. And it turns out we can do that as long as we conjugate it.

So we've moved the cue from the left side of the dot to the right side of the dot and just conjugated it. And it's easy to prove that. We just multiply by cube, conjugate, and so on. So we're going to use unit quaternions to represent rotation. And we'll see right now how that's related to the other notations for rotation, but we're dealing with vectors. So what about vectors?

Well, we'll need to have a quaternion way of representing vectors. And of course, it's obvious. We just leave out the scalar part. Conversely, if we wanted to represent the scalar, we could just leave out the vector part. OK, so we're going to convert our vectors in 3D to these funny quaternions, purely imaginary quaternions, manipulate them in that space, and then bring them back into 3D.

For this special case only of these types of quaternions, there's lots of interesting properties. First of all, the conjugate, of course, is just negating it, because conjugate means negating the imaginary part. The dot product is just the dot product. So this is the dot product of the quaternions. This is the dot product of the corresponding vectors. And since the scalar part is 0, it's obvious that this is the case. So we can easily

Compute dot product. If we multiply two of these special quaternions, we get this funny thing. It has the dot product in it and the cross product. And this is why some critics of this notation called it a hermaphrodite monster. It's got both sexes built into it, dot products and cross products. And they thought this was a disadvantage. For some purposes, this is actually an advantage.

Then if we take the product of r and s and take the dot with t, we get the triple product, rst. And the triple produc is often pretty useful, aside from computing volumes or something. And again, this is very simple to prove. Once you have a 0 scalar apart, then all you're going to get is the vector part of t dotted with this thing, r cross s. And of course, that's just r cross s dot t. That's the triple product rst and so on. So that's how we represent vectors.

OK, scalars-- yeah, we can do that. Finally, we get to rotation. So how do we do rotation? Well, it turns out that simple product of a quaternion with another quaternion won't do it, because it takes us out of 3D into 4D. We need an operation that takes us back into 3D. And so this is a little bit like you can do rotation in the plane by doing some operation that takes you out into 3 space. But then you have to have another part of the operation that takes you back into the plane, now, in a rotated form.

So this is like that. If we multiply two quaternions, we're basically doing an operation of rotation in 3 space. And now, we need to find a operation that doesn't undo that but takes us back into 4 space. And this is exactly what happens here.

So we take our vector, r, turn it into a quaternion with 0 scalar part. Then we pre-multiply by q and post-multiply by q conjugate. And magically, we're back in the real world so that this quaternion, r prime, has a 0 scalar part. And so this is how we're going to do rotation.

Now, there are many different ways of analyzing this. One is to use that trick we had of representing quaternion multiplication as multiplication of 4 by 4 matrices and the four-vector, right? So we take these first two and turn them into that. And then we take the second multiplication. And this time, we turn the q conjugate into a matrix, this matrix over here.

OK, so this operation is actually equivalent to that operation. And what is this? This is a 4 by 4 matrix. It's a product of two 4 by 4 matrices. And if you multiply them out, you get this. And this-- the most interesting property of this is that the first row and the first column are mostly zeros.

And the result is that if you have a vector in 3D, which has a zero part-- scalar part and multiply it by this, there'll be no 0 scalar apart, right, because you're multiplying this by 0. And then you're adding all of these zeros multiplied by whatever. And you get 0.

So all this has proved so far is that this operation will, in fact, get you back into 3D. So and this-- by the way, this submatrix is-- and this actually is our 3 by 3 orthonormal rotation matrix. OK, so that's the equation we're going to use for rotation.

And this is the scalar part. We can compute the scalar part. And if the scalar part of the original vector is 0, we get 0 for the scalar part of the new vector. The vector part can be computed this way. So if you don't want to jump into the ocean of quaternions, you can do everything using scalars and vectors. Here's the formula, right? So just dot products and cross products-- very easy.

When we use this operation, we can easily-- well, with a bit of algebra, we can prove that it preserves dot products, right? You just take this formula, apply it to r, then apply it to s, take the dot product. And you get r dot s. And so it preserves dot products.

Similarly, it preserves triple products. Remember we had this special form that this multiplication gave us a triple product of the vector, the underlying vectors? Well, it preserves triple products. So that's it. It's a rotation, right? It preserves dot products. It preserves triple products. So that's good. We don't know yet what rotation, but we know it has to be a rotation, because it preserves length and angles. And it preserves handedness.

And then the other thing we were talking about was not just rotating a vector but composing rotations. We wanted that to be easy. Well, so suppose we first rotate-- we take our vector, r. We rotate it using q. And then we take the result, and we rotate it using p. So we write this.

And because these operations are associative, I can rewrite it this way. And that tells you, oh, composition of rotation is just multiplication of quaternions-- it's very easy and actually computationally efficient, as well, which is very different from axis and angle notation or Euler angles, where it's very hard to compose rotations.

So then we need to figure out what rotation is it, or if we have specified rotation in some other form, how do we turn it into quaternion? Well, this is the formula that we just sort of derived on the previous page. And here's Rodrigues' formula from a while back.

And then if we identify corresponding pairs, first thing you notice is that q, the vector part, is parallel to omega. So the vector part of the quaternion is in the direction of the axis, which sort of makes sense. And we can actually prove that easily. If you plug q in here for r, what happens? Well, this part drops out, because a a cross 2 is 0. And then this becomes q dot q times q. And this becomes something times q.

So the whole thing-- both of these are vectors in the q-direction. So r prime is going to be in the q-direction-- so very easy to show that q becomes q. And that's Euler's axis theorem. So it's very easy to see that the vector part is parallel to the axis of rotation.

We don't know yet how long it's supposed to be. And we don't know what the angle is. But anyway, leave out some of this-- conclusion is this is our representation-- our conversion. So if we know axis and angle, we can compute the quaternion. So that's one of the conversions between the-- we have eight different ways of representing it. So they're 8 times 7-- 56 different conversions we could possibly do, but we're not going to do them. So this is one.

The other one we saw earlier, where we produce the orthonormal matrix. That's another important one, because we use orthonormal matrices all the time. We need to be able to go back and forth between axis and angle, which is this formula, and between the orthonormal matrix, and that was that previous formula.

Now, so there are two things about this. One of them-- this is a unit quaternion. We easily proved you take its dot product with itself. And you get cos squared theta over 2 plus sine squared theta over 2. So that's one thing. So we're not talking about arbitrary quaternions, just unit quaternions.

And then the other thing that's a little bit annoying is that minus q is the same rotation as plus q. Why? Well, imagine you plugged minus q into this formula. The minus is just-- disappear. So what does this mean? It means that if we think of the sphere as representing the space of rotation, opposite points are actually the same rotation.

So it's very nice to think of a sphere in four dimensions as the space of rotation. And then we can produce ways of sampling that space and taking integrals and taking-- computing averages and so on. But you have to keep in mind that you really only want half the sphere, because the opposite side, the reflected side represents the same rotations. And [INAUDIBLE] points are identified.

So we're going to use this in photogrammetry and in particular, in absolute orientation. We already talked about that. And this is from last lecture. So we got points in space identified, measured in two coordinate systems. We want to know the coordinate transformation between those two systems.

And it's dual to the other problem, where we have the same coordinate system. But we have two objects-- or we have one object moving. And we modeled it this way, where coordinates in the left system are rotated, and it's translated to produce coordinates in the right coordinate system.

And we want the best-fit rotation and translation, meaning we want to minimize some error. And what we're given are a bunch of corresponding points measured in the 2 coordinate system-- left-- left and right. And then we talked about this mechanical, this physical analog that we want to make these errors small.

And so one way to do it is to have these springs, which have energy proportional to the distance squared. And the energy-- total energy in those springs is the sum of squares of errors. And the system wants to minimize that. So that's the physical model of how it finds the rotation and translation.

So then we did this. So this is the error term, right, because if I take the left coordinate and I rotate and translate it, I get this. And that should be equal to the right coordinate. And I subtract them. I get the remaining error. I square that. I add all of those errors.

Then first step-- let's find the translation. And we did this last time. One way to do it, which is not the way we did it in class, is just differentiate with respect to the translation, r0. And this notation-- this is the norm, which is really the dot product of this thing with itself. If we differentiate it, we get twice this. And we set that equal to 0. And then we can split. So we have a sum of three terms. We can split that into three sums.

So for example-- of course, we get rid of the 2. That's not very interesting. And we get this. And final result was just that the translation is what takes the centroid in the left system after rotation into the centroid of the right system. So that's kind of intuitive and nice. It says that whatever your transformation is, it should map the centroid of this point cloud into the centroid of that point cloud.

And one of the nice features of it is that you don't even need correspondences. You just compute the centroid of those clouds. So for this part, you don't need to know which point in the cloud corresponds to which point in the other cloud.

And of course, right now we can't get the answer, because we don't know r. But this is the formula that we're going to come back to at the end to get the translation. And so at that point, we can move the origin to the centroid and add a dash to the r's, indicating they are now measured with respect to the centroid.

So when we do that, the formula simplifies. Now that we're minimizing this expression, where these r prime and R r prime and rl prime are defined in terms of coordinates with respect to the center, we subtract out the centroid-- subtract out the centroid there.

And OK, then we take this. And we multiply this out. Again, the non-squared is just the dot product of this thing with itself. So think of it written out that way. Then when you multiply out, you get four terms. You gather them up, and you get this.

And here, we use one little trick, which is that we noted that the length of a vector is not changed by rotation. So over here, we really get r times the left vector squared. But we know that after rotation, it's the same length. So we just replace it.

OK, and at that point, these two things are fixed. They're given by the data. They're not going to depend on the rotation you pick. So forget about them. We need to focus on this middle term. It's the only one we have control over. We can change r. But it has a minus sign. So we're trying to minimize something. So that means we want to maximize this term. And that's what we got to last time.

And I was explaining it in terms of the sea urchin analogy, where we've got these spines. And we're trying to get them to be as lined up as close as possible so that their dot product is as large as possible. The dot product of corresponding point-- now, we need correspondences.

And this is the classic problem in spacecraft attitude control. You have the direction to stars in your cameras. And you're trying to line them up with the catalog directions. And of course, you want to make the-- and it's a good analogy, because in this case, we don't care about the length. We don't have the length. We don't know-- from the camera, we have no idea how far away the stars are. And we don't care.

And if we move sideways, actually that direction doesn't change, because in comparison to the distance to the stars, our own motion is microscopic. So this would be the thing you want to maximize in that case. And but how do you do it?

Well, we're sort of tuned to using calculus to solve all these problems-- just differentiate with respect to the unknown transformation. But what does it mean to differentiate with respect to r? Now, I'm sort of making fun of it. But you can actually pursue this further.

What you need to do is not just differentiate with respect to r but impose all of those constraints that I talked about. And you can do that. It just gets messy. You have to impose that r transpose r is the identity. And you have to impose the determinant of r is plus 1. Now, that's the hard one.

OK, so let's not do it that way. Let's use our new fangled quaternion notation. So here, we have the dot product of two vectors. And we can express that here as the dot product of two quaternions with 0 scalar part.

So we map this r li prime into a quaternion, r li prime. And we map this R r i prime into this quaternion. And all you've done is add the 0 scalar part. I've also flipped them around, because dot products commute-- so can do that.

OK, and this is where I used that trick. Remember when I said that sometimes you want to move something from one side of a dot product to the other. Here, I've taken this quaternion, flipped it to the other side. But I had to conjugate it. Now, it's already conjugated. So conjugate of the conjugate is the thing itself. So I get this expression.

And then I make use of the property that I can write this quaternion product as a matrix times vector. And I can write this as a matrix times vector. So I get that. And then I make use of the fact that a dot product is just the left thing transpose multiplied by the right thing.

And then finally, I can pull out the q. So this is important. I can separate out the q on both sides. So it doesn't vary with i. And so I end up with this product. So this is where all my data is. All my measurements are buried in here. This is a 4 by 4 matrix that I get from my measurements.

And now, all I want to do is make this thing as large as possible-- well, not quite, because I can make this as large as I like by making q large, right? So I have to be careful to impose that constraint. And so quaternions are not a redundant representation, because they have four numbers. And you have to impose this extra constraint.

OK, and then there are different ways of doing this. So first of all, let's see. What is N? Well, N is, again, got from the data. And there are two ways to do this. If you know about Lagrange multipliers, you can add another term that imposes the constraint. So the constraint is that q dot q equals 1, or minus 1 minus q dot q equals 0. So you can impose that way. And we'll do it another way that doesn't involve needing to know about Lagrange multipliers.

Anyway, in either method, we can now differentiate with respect to q, which is amazing. We can differentiate with respect to rotation. Of course, in the case of translation, no one's surprised you can differentiate with respect to translation or position. But for rotation, that's quite something.

If we do, we get this. And this is just using the ordinary rules for differentiation of a dot product and differentiation of a matrix times a vector, which by the way, are in the appendix of the book, which by the way, is on Stellar. So if you need to refresh your memory about differentiating with respect to a vector, it's all there.

And so what does this say? Well, this says that Nq equals lambda q. Well, that should remind you of stuff we've done before, right? It's an eigenvector. So q has to be an eigenvector, and lambda is the eigenvalue. And since we want to make this as large as possible, we pick the eigenvector that corresponds to the largest eigen-- which is sort of unusual. In the past, we've always wanted to minimize something. We wanted the smallest eigenvalue here because of that sign flip. We want the largest one.

And what is it the eigenvector of? It's a 4 by 4 real symmetric matrix, which is constructed from the data. And so a couple of notes on that-- so one of them is we said for 2 by-- for a 2 by 2 matrix, we actually explicitly gave the eigenvalues and eigenvectors, because we know how to solve quadratics.

And I mentioned last time that for 3 by 3, it can be done, because the characteristic equation is a cubic polynomial in lambda. And someone knows how to solve cubics in closed form. By the way, this used to be a big game, particularly amongst the Italian mathematicians. They would-- these problems were a real puzzle to them. And they were very proud when they solved them correctly so.

But there wasn't the same sort of modern publication and tenure and whatever. It's like you've solved this tough problem. And you're very proud of it. What do you do with it? You don't want to just give it away. And yet, you want to be able to say later, you know, I got that 10 years ago.

So they've come up with all these weird ways of coding and coding their solution and like in a story or as a sort of parable or some sort of other mathematics. And then later on, Ferrari, who was one of the guys who did this, could say that, oh, I solved this years before Cardano, who is another guy, because look at that poem I wrote. It tells you how to do this.

So anyway, they discovered how to solve cubics. Here, we're going to have a quartic, right, because it's 4 by 4. It's going to be fourth order. And guess what. They have closed-from solutions. And those Italian mathematicians and others figured out how to do it.

Basically, how do you do any of these things? Well, the usual trick-- you try and find a way of reducing third order to second order, because you know how to do second order. And quartics-- first step is reduce it to a cubic. Solve the cubic, and then the rest is easy.

And so can you solve fifth order? No, OK, so it's good that some of you know that that's it. You can't-- now, of course today with computers, we can always numerically solve any of these. But if you want to have a, quote, closed-form solution, then you do need to know that you can solve 1, 2, 3, 4. But that's it. Then higher order, you can't solve in closed-form using addition, subtraction, multiplication, and square roots.

OK, so this matrix down here-- it look-- is more familiar to you than N. That's because this is simply the dyadic product of the vector in the left coordinate system and the vector in the right coordinate system. So this is a 3 by 3-- each of these is a 3 by 3 matrix that's a dyadic product. And you just keep on going, stepping through your point pairs and adding them up. And you get a 3 by 3 matrix, which is not symmetric. And so it has nine independent quantities.

And let's think about N. Well, N is 4 by 4. So it has 16. So how does that work? It turns out that N is rather special. It turns out N is symmetric. And it turns out that it's not that easy to prove. The one mistake in this paper was that I think I said something like, it's obvious that it's symmetric. But it takes-- anyway, it's symmetric.

So that means 4 by 4 symmetric-- we've got 4 on the diagonal, and then we have 6 more-- no, 10-- we have 10 independent-- so it's not 16. But it's 10. But it's still too many, right, because we only have 9 in M. Well, it turns out that this matrix is very special, because the determinant-- the characteristic equation up here has this important property, that that cubic term is 0, which actually makes it easier to reduce it to a cubic.

And so N is a symmetric 4 by 4 matrix with a trace of 0. So that means we've got 10 minus 1 is 9. So now, it matches. M has 9 independent values. OK, we don't need to really know-- and if you're actually going to implement this, you might want to know this, but just do know that you can compute all the coefficients of the characteristic equation. And then you go off and solve your quartic.

So the main application is going to be absolute orientation, but it also has lots of other applications that we already talked about. And if you go over to Draper Lab, where they do all this spacecraft control, you'll find people, who know about quaternions, because that's what's used there.

Desirable properties-- so we have this table. Let's see how we're doing. Ability to rotate vectors-- yes, we-- qr q conjugate. Ability to compose rotations-- yes, p times q is the composition. Intuitive, nonredundant representation-- well it's almost nonredundant. It has four numbers to represent 3 degrees of freedom. And but the redundancy is very simple. It's just the unit vector-- unit quaternion.

Is it intuitive? Ah, one part-- the vector part is in the direction of the axis. That's sort of intuitive. Computational efficiency-- we haven't talked about that yet. How is it compared with matrix operations? Can we do things like interpolate orientations?

Yes, suppose that you have-- your artist has the ballerina doing a turn and got it-- got her in this position and then in that position rotate it, how do you do all the inbetweening? Do you take the initial rotation matrix and the final rotation matrix-- take some sort of weighted average?

Well, that's not going to be orthonormal. So here, it's trivial, right? You just take that angle. Suppose that the rotation is about an axis through an angle of 90 degrees. You just split it up into however many frames you need and compute the quaternion correspondingly.

We can take averages of a range of rotations. If you want to know the average loading on the seat belt as your car is tumbling in all possible orientations, you can easily do that. And we can take the derivative as we saw. So we can do optimizations, least squares. If there's no closed-form solution, we at least have a way of sampling that space of possibilities in the uniform or random way.

Oh, OK, well, let's see. How late is it? Well, let's do a couple more things. OK, so this is where we're going to go after absolute orientation. This is relative orientation. And this is kind of more-- so absolute orientation is very important for photogrammetry, making topographic maps from aerial photographs. But in terms of understanding human vision, one of a dozen depth cues is binocular stereo. And so relative orientation there is more important.

And so that's the problem of, we've got two eyes. And the eyes don't get depth. They only get directions. So we have the directions in each of those coordinate systems to points out in the world. And I show in 5. And it turns out 5 is the minimum number you need to solve this problem. And your problem is to find out what that baseline is and what the relative orientation of this coordinate system is relative to that. In other words, we have a translation. And we're going to have a rotation.

So in that respect, it's the same problem. It's just that our data now is not as good. But-- in absolute orientation, we actually know 3D coordinates of all of these points. And that makes it possible to get a closed-form solution. Here, we only know the directions. So that's the problem we're going to solve after absolute orientation.

Then if you're trying to describe the kinematics of a robot manipulator, well, these days people are lazy. You just buy it. And the software does all that for you. But in the old days, we had to walk in the snow. There was no bus. Well, in the old days, you actually had to understand the geometry of these devices. And that means that as you go through each link, you are doing a translation and a rotation.

And so if you want to avoid problems with coordinate systems lining up, particularly in the wrist, then quaternions are a good way to represent the rotation. So in the wrist, for example, you can imagine that if this part now comes up, there will be a time, where this axis in here is parallel to that axis. And you've lost a degree of freedom.

So computational issues-- now, I guess these days computers are pretty fast. So we don't worry about this as much. But if you have to do this for every point in some object in your graphics representation, this could be an issue.

So let's see how expensive it is. So again, there are two things we want to do-- compose rotations and rotate vectors. So for composition, we know it's just multiplying two quaternions. And if you do it the naive way, just following that formula, you get 16 multiplies and 12 adds. And if you do it with an orthonormal matrix, of course, it's a 3 by 3 matrix product, which is 27 multipliers and 18 adds. So we win-- definitely win here.

Let's go to rotating a vector. So we use this formula. And we can expand out the part we want, which is the vector part-- 22 multipliers, 16 adds. Compare the matrix-- oops, 9 multiplies and 6 adds. So here, we lose. But this can be rewritten in this form, which has the advantage that you compute the qr once, and you use it in two different places. And you get it down to 15 multipliers and 12 adds.

And this is naive stuff. People that get serious about this go further. For example, with a 3 by 3 matrix, they don't represent it as 3 by 3 matrix. At most, you take the first two rows, because the other one's implicit. It's a cross product. And so both-- so you could have a competition, where you just keep on going on both sides. And they end up being similar, but with the advantage for composition going to quaternions and the advantage for rotating vectors going to orthonormal.

OK, how about renormalizing? So one of the things that happens-- suppose you have a graphic program that's trying to do some motion sequence. And you're composing small incremental rotations. And if you represent them, let's say, as orthonormal matrices, you can imagine that because of the limitations of floating point arithmetic, that errors will accumulate. And after a while, your 3 by 3 matrix is no longer orthonormal. And so you'd want to sort of square it up.

And that can be done. But it requires doing this. You take your matrix, and you multiply-- you take its transpose, multiply by M. And you take the inverse of the square root of the matrix. So right there I'd imagine that you may not know how to do that. But it can be done. But that just suggests that this is not an operation that you'd normally want to do. It's expensive.

With quaternions, sure enough, you take the unit quaternion, and you multiply it by another unit quaternion and so on. And because of the limitations of floating point arithmetic, after a while it isn't a unit quaternion. How do you square it up? Well, trivial-- so this is one other small difference between them.

So just about at the end-- so we talked about the space of rotation being the 3 sphere, unit sphere in 4D with antipodal points identified. And it also happens to be identical to the projective space p3 although we don't use that. And it's not as handy.

Then sampling regular and random-- so how do we sample this space? Well, in 3D, we can easily come up with a regular sampling of space or even the random sampling. But it's not so obvious how to do it here. And one way is to find regular patterns.

If we're-- say we want to sample a sphere regularly. That's not so easy either. Certainly, latitude and longitude is not going to do it. But one thing we can do is project a polyhedra onto it, because polyhedra are completely regular.

So if we project, say, a icosahedron or a dodecaicosahedron, which is a soccer ball-- anyone here know about soccer-- then we end up with a pattern on the sphere that is regular. But there are only some special ones, like regular solids. There are only a few of them. Well, similarly here, except now, we're in 4D, and the corresponding things are rotations rather than polyhedra.

And so it turns out that if you look at the rotations of polyhedra, they will give you a nice, even sampling of that space and the rotation groups-- so the tetrahedron has 12 rotations that will align it with itself. The cube and the octahedron have 24. And the icosahedron and dodecahedron have 60.

So these give you a perfectly uniform spaced sampling of that space. Unfortunately, that's not a very fine sampling, right, because 60 in the four-dimensional space. So there are tricks to give you a finer sampling. But they all start with those simple ones.

And just while I have the slides here, if you pick the coordinate systems right, you get very simple expressions for these rotation groups. So each of these is a rotation. This one, of course, is the identity rotation-- does nothing.

This is the rotation about the x-axis by 180 degrees. And this is about the y-axis and so on. So this is the group for the tetrahedron. They provide a regular spaced sampling of that space. This is for the hexahedron and the octahedron.

And then the prize is this one. This is the rotation group for the dodecahedron and an icosahedron, where these a, b, c, d's are given by these expressions up there. These should look familiar. This only works if you line it up nicely with the coordinate system. Anyway, this would be a sampling. So you could try these 60 orientations for your search or your averaging. And then you can interpolate further. OK, well, we've run out of time. So that's it for today.