PROFESSOR: All right before we get started, I just wanted to say goodbye to Eric Joisel, who died on Sunday, sadly. He's a master origami creator, and I encourage you all to come see "Between the Folds" on Tuesday, showing Tuesday evening, because there's a great interview with him, and it really captures his spirit. And sadly, I never met him in person. The last I corresponded with him was to get permission to show these photos in class. So that sucks.

Today is about rigidity again.

And in particular, something called infinitesimal rigidity. So last class was mostly about generic rigidity.

Today we're going to talk about a variation, although related to generic rigidity. And this is going to be a very useful tool. It's a lot easier to work with algorithmically. It still captures essentially generic rigidity, but in a different way-- in a linear way. And it's going to be super useful.

So the idea is we think about rigidity-- does a linkage move or not-- to the first order. First order means first derivative. OK, so I'm going to do some analysis stuff-- calculus, which you should all know, because is the General Institute Requirement. Don't worry, you don't need to know much.

So in particular rigidity, is really about the lack of motion. So let's define the idea of infinitesimal motion, or first-order motion. This is, again, suppose we start somewhere. And somewhere would be a linkage configuration.

There's a linkage. There's a graph, with edge lengths plus configuration maps every vertex to a point. I'm going to assume we're working in the plane for now. But really this works in-- I'll say d dimensions.

So this is what I want to define. I'm going to say that it is a valid-- this is sort of an informal notion-- a valid first derivative of a motion. We've already defined motions.

I want to take such a motion, essentially, take its derivative with respect to time, And evaluate that derivative at time 0. This is a fancy way of saying the following picture-- I'd like a motion, which is some path in the configuration space-- remember, configuration space is some hugely dimensional object, d n dimensions-- and a point in that space corresponds to a configuration. We have a motion is some path in that space.

And motions are hard to think about. They can be really complicated. So I just want to say, well, how do you get started moving? What direction do you go in in a linear sense, to the first order? What's the derivative? What's the tangent right at the beginning?

And then this notion will, stated in terms of motions-- you can think about this in the absence of a motion. Say, well, yeah, I don't really know whether this curve exists, but let's just see, does this initial starting direction exist? And that's actually a little weaker than the whole motion existing.

If the motion exists, surely you can take this derivative and evaluate it. It's not totally obvious, but it's true. You need some smoothness. But if there's a way to get started moving, that doesn't actually mean you could actually move.

Why don't I give a more formal definition, and a more useful one-- but that's the intuition. So another way to think of an infinitesimal motion is, instead of saying how each vertex moves in d dimensions, you just give a velocity vector for each vertex. For every vertex v, we're going to get some velocity vector d of v-- that's direction or derivative of v.

And with motions, we were supposed to preserve edge lengths. So here we'd like to preserve edge lengths, but that's actually kind of hard to do. If you take an edge-- so I have two vertices, and I assign some velocity vectors, something like this-- as soon as you move a little bit in that direction, the edge length won't be quite the same.

But it'll be the same to the first order, so this is going to be a little subtle. And in the book, we take the derivatives. We work out all the details. But I'll just tell you what the condition is, because ultimately all we need is this condition.

I'm just defining an infinitesimal motion to be a velocity vector for every vertex, such that this property holds. This is a dot product between two vectors. If you have two vectors, a and b, you take their dot product, this is just something like this. You take the x-coordinates. You multiply them together. You take the y-coordinates. You multiply them together, and so on. Add them up.

And we're going to need some linear algebra. That's one useful linearity thing. So these are where the configuration places vertex v and vertex w. So this should be true for each edge v w on the graph.

This is a vector C of v minus C of w. Let me give some labels. Let's say this is w, and this is v.

Let's say these are the points that the configuration places w and v at. Then C of v minus C of w is this vector. It's a vector.

It doesn't really have any space. It's just been translated around. It's a vector that would point from w to v in the configuration. That's the left part.

The right part has no real intuitive meaning, but it involves this thing-- some velocity vector for w-- and the velocity vector for v. So this is some relation on this vector and these two vectors. It's actually a very intuitive relation. Let me tell you what it means.

Basically, we want to understand how this length changes to the first order. So this guy's moving to first order in this direction. In fact, it's moving on some curve, and the initial tangent is that direction.

And to understand how that motion changes this edge length, turns out the right thing to do is to project this vector onto this one. Makes sense. So as this guy moves in that direction, This is the first-order change. This distance here is the first-order change of this length as caused by this vertex moving.

OK, that projected length is actually very simple. This is again, some linear algebra stuff interpreted geometrically. This is a dot product in d of w-- that vector-- dot product with C of v minus C of w.

It turns out dot product corresponds to projection. So I'm taking this vector. I'm projecting onto this one. That's how you'd write it algebraically.

And same deal over here. If I project this vector onto this one, that's going to be the first-order change to this edge length caused by this vertex moving in that direction. So this is d of v dot C of v, minus C of w.

So if I want this edge length not to change at all to the first order, then this projected length should equal this projected length. And that's why I drew it this way. So it turns out this is a valid motion, because that projected length equals that one.

And in general, we want this thing to equal that thing. And if you rearrange terms, that's the same thing as this property. I just take this one minus this one, and set it equal to zero, and you get that.

OK, so this is some intuition. Maybe it makes sense, maybe not. In the end, this is all I care about.

And it's useful to realize that it's a first derivative of a motion, because what that implies is that if there's a motion-- a real, honest to goodness motion-- then there's an infinitesimal motion. I'm going to abbreviate that "inf motion." Infinitesimal is pretty long. Infinitesimal means smaller-- very tiny-- smaller than anything, I think is the literal meaning.

So this is useful to us, because the contrapositive says if a linkage is infinitesimally rigid-- which just means if there is no infinitesimal motion-- then it's rigid. So these are the same statements. This you prove just by taking a motion and taking its derivative. This is identical to that.

So this is useful because if we can ever find something and show that it's infinitesimally rigid, then we've determined that it's rigid. Question?

STUDENT: [INAUDIBLE] dot product [INAUDIBLE] at a left right turn is 0, or that the terms are at 90 degrees to each other?

PROFESSOR: Yeah, this dot product basically means these two vectors are at 90 degrees to each other. That's another geometric meaning. I don't have a geometric picture. Maybe there is one, but I have trouble thinking about d of v minus d of w. That's not an intuitive thing to me, but it's something.

And we're saying that that vector should be perpendicular to C of v minus C of w. However you want to think about, I like to think about it this way, but they're equivalent.

All right, so how do we tell whether something's infinitesimally rigid? Because somehow that's useful to us. Well, there's a saying that everything is linear to the first order-- sort of a tautology. And all you need to observe is that this constraint is linear in what we don't know.

So I want to know is there some set of velocity vectors d of v, and d of w, for all the vertices, so that these constraints hold. But I already know C of v, and I know C of w. This is just where are these vertices currently. And I want to know can I get started moving?

So these are my known quantities, and these are my unknown quantities. This is just some linear thing. And the dot product-- I mean, this is all just constant. I know what they are. So this is a linear equation.

The fancy way to write this is as a matrix equation, if you want. So this is the idea of a rigidity matrix. So we have a whole bunch of these constraints, one for every edge.

And whenever you have a bunch of linear constraints, you can write them as R-- some big matrix-- times the velocity, or call them d's equals 0. So I can absorb this into a matrix. If you're used to that, great. Otherwise, this is what it looks like.

My matrix R is going to have a row that has a whole bunch of zeroes. And then at some point, it has basically various versions of this vector. But because this is one equation, but in two dimensions, there's two parts to all these vectors. Two dimensions that'll look something like this-- why did I change the order? I think it's right.

OK, this is not very amazing. It's just there are going to be four terms in two dimensions in this equation, because we've got d of v times this vector. So there's a x-coordinate vector, and the y-coordinate vector, and in d dimensions, there will be d of them. Here in two dimensions, I've got the x-coordinate of the vector, and the y-coordinate of the vector.

So this will end up getting multiplied on the right hand side by d of v. So the two v rows, and this part will end up getting multiplied by d of w. yes, so these guys switched around. It's now w minus v in order to implement that minus sign.

So whatever-- the point is you get a matrix. And what we care about, what we're trying to solve for, is d. We know this entire matrix, because we know c. We know the configuration.

We just want to know is there some set of velocity vectors such that the rigidity matrix times this is equal to zero? This is called the set of d's i for which this is possible is called the-- anyone know? Good old linear algebra-- it's called the null space of this rigidity-- null referring to that 0, or the kernel.

So the set of all infinitesimal motions is the null space of that matrix, also known as the kernel of the matrix. And so what? Well this lets us use a bunch of theorems in linear algebra.

So one fun fact is that the dimension of this space-- the space of all infinitesimal motions-- seems like a useful thing to know. We want to know how many-- this is basically the number of degrees of freedom in infinitesimal land, which is not quite reality, but it's close. This is called the nullity of the matrix.

That's just the definition, but there's a fun theorem called the rank nullity theorem from linear algebra. Which says if you have a matrix, you take its rank-- which is another quantity-- you take its nullity, and you add them together, it will be the number of columns in your matrix.

Now in our case, the number of columns is d times n. We have n vertices down here, and each of them has d coordinates. So for us, this is d times n.

So this is like a conserved quantity. What we care about is the nullity. This is the space of motions-- infinitesimal motions. And so consequently, we care about this thing called the rank.

And if you've ever seen linear algebra, especially linear algebra algorithms, the bread and butter business is computing the rank of a matrix. This can be done in polynomial time. So this is good news.

And intuitively what it corresponds to is the number of useful edges. And this thing is the number of degrees of freedom. So it turns out these will always be equal to d times n in summation.

And what we care about, infinitesimal rigidity, means that-- or let's say it's infinitesimally rigid if and only if the nullity of the matrix is the space of rigid motions, which I have to look up. In general, in d dimensions, it's this-- d plus 1 choose 2. In two dimensions, there's three rigid motions. You've got the two translations and the rotation.

Those we're not getting rid of. They still exist in infinitesimal land. So in two dimensions, ideally your nullity is-- if you want rigidity, your nullity should be 3.

And what this is telling us is then the rank of the matrix should be d n minus that. In two dimensions, that is 2n minus 3. So this is telling us something we already knew, but essentially for free, which is kind of fun, if you know all this linear algebra. Which admittedly, even if you've learned linear algebra, that may not be the first thing you care about in that context, but for rigidity, this is a really useful way of thinking about things.

You've got all these columns-- d times n of them. You want to obliterate all the degrees of freedom you can, and leave just three of them in two dimensions-- d times d plus 1 over 2 in d dimensions, and you do that by adding useful edges. And useful edges here means that the rank is high, which essentially means none of the rows in this matrix cancel out with each other.

So it's another way of thinking about essentially everything we've seen. And if you know linear algebra, it's useful. Otherwise, I'll tell you how we can use it.

OK, one mathematical thing we can do is give another definition of a generic point set. Last class we talked about one definition of generic, which was that you forbid all non-trivial polynomial or algebraic rational equations on your points, which is very messy and infinite and hard to think about. This definition will be easy to think about if you're used to linear algebra.

So for some, it's an improvement. For others, it might be the reverse-- whatever. But at least this definition is finite, which I like.

OK, for this definition we need the notion of a minor. So we have some matrix. Actually, our matrices are pretty rectangular.

And I just choose, let's say k columns from the matrix-- let's say 3, and I choose k rows-- the same number of rows and columns. I look at those elements in the intersection. That's a square matrix. That's a minor-- that 3 by 3 square submatrix is called a minor.

So there's a whole bunch of minors, but it's only finitely many. It's exponential. It's not easy to check.

You take all the minors of your rigidity matrix. And here I'm imagining the complete graph, so there's tons of edges. There's n choose two edges. So this'll be independent linkage.

You could do it specific to the linkage, but it's better to do it for the complete graph.

Then we take the determinant of that matrix, which I won't define. It's a number associated with that matrix.

And if it's non-zero-- now remember, rigidity matrix is defined as a function of C. It depends where you put the vertices. So what I'm saying is if it's non-zero for some choice of C, then it should be non-zero for this choice of C.

And if you find such a configuration C-- or I guess we call this a realization C-- then that C is generic. So if I gave you a specific configuration, you could, in principle, in finite time-- it won't be efficient-- you could check all of these things. And say, oh, I compute out, I work out what the determinant of that submatrix is, that minor.

It's some algebraic function. If it doesn't cancel out entirely, if it's not zero trivially, it's not zero always, then I just evaluate it at this configuration. And if it's non-zero there, good. That's generic. So it's not useful in a computable way, but it's a definition. It's a little bit nicer.

And now there's a bunch of fun facts about infinitesimal motions and rigidity, and the relation to the generic case. So there as we have before, this definition is not quite the same, but it's effectively the same as the last one. Almost every configuration is generic.

So maybe I should say realization-- whatever. I take some point set-- I take some vertex set. I assign random points to each vertex. Then with probability 1, it will be generic according to this definition. That's really obvious if you know linear algebra. That's a good thing.

But here's what's really cool-- if we take any of these generic configurations-- so just fix one-- rigidity is the same thing as infinitesimal rigidity, is the same thing as generic rigidity from last class. Wow, this is cool, because if I care about any one of these three things, I just have to pick my favorite generic point, which I can do randomly. I can just take a random point, and if I can evaluate any one of these three things, I learn about all the others.

So if you want to tell whether a 3D graph is generically rigid, you take a random realization in 3D. You compute the rank of this matrix generalized to the 3D case. You see is it 3 n minus 6, and that will tell you whether that generic realization is infinitesimally rigid. And if you chose it right, which you will do at probability 1, then that will tell you whether the graph is generically rigid.

So when I said last class generic rigidity in 3D is hard, it's hard in the sense we don't have a deterministic algorithm, but randomized-- super easy. Choose a random configuration, check the rank of the rigidity matrix-- done. So that's kind of cool. And that's why this infinitesimal rigidity stuff is so important, because it's so easy algorithmically. Things become a lot easier.

Let me draw you one example just to show what happens at non-generic points. I've actually drawn this example before. So here's a non-generic point. These three points are collinear. And you can work out there is some minor that is 0, but shouldn't be.

And this thing is rigid, because we have rigid triangle here. We have a rigid triangle here. So these two points are pinned relative to each other.

And then this thing is taut, so it can't move. But to the first order, it can move. And it's allowed to move straight up or down-- anything perpendicular to this segment.

That's always true. If you take a segment, and you move one vertex but not the other perpendicular to the segment, that's a valid infinitesimal motion. Why? Because if I project this onto the segment, I get 0. So this length is not changing to the first order.

And this is really necessary. To the first order, you can't really distinguish this guy actually moving up, and therefore getting slightly longer versus this guy moving along a circle centered at that point, which is a valid motion. To the first order, they look the same, because you're going straight up or down in first derivative.

OK, here, to the first order, you can't realize the fact that if you're going to go on this circle and you're going to go on this circle, you get a contradiction. But it's pretty accurate. And if you perturb this example, infinitesimal rigidity will give you the right answer.

This is an example where infinitesimal rigidity is a little bit weaker than-- no, infinitesimal flexibility having infinitesimal motion is a little weaker than having a motion, but the reverse holds. If there's a motion, there's always an infinitesimal motion. If you're infinitesimally rigid, you're always rigid. That example is rigid, but not infinitesimally rigid, so it's sort of strict.

OK so far? That's the more technical part of the lecture. Question?

STUDENT: Will [INAUDIBLE] be square matricies?

PROFESSOR: The rigidity matrices are almost never square. Good question. The minors are going to be square, because we told them to be.

But I think typically, in the ideal setting, we're going to have d n columns-- so let's say 2 times n columns in two dimensions. And we're supposed to have 2n minus 3 rows. So it's close, but in general, they should differ by about this much. So they're almost square, actually. Glad you asked.

STUDENT: You can add [INAUDIBLE].

PROFESSOR: You could always add extra edges and make it square, yeah.

STUDENT: [INAUDIBLE].

PROFESSOR: Well, no. So that's another good question. What happens if I add more edges than this many?

The rank just won't go up. Any edges you add will add redundant constraints, and so the rank can never go above this value. You need at least this many edges to get there, and we know we can get there with that many edges, but only in the minimal rigid case.

You could add more edges. You could make it a super-tall matrix if you want, but you're not going to increase the rank. All right, let's use this for something more fun. So maybe go over here. I don't need the rigidity matrix any more.

Tensegrity-- so this is in some sense an extension of rigidity theory to a whole other theory called tensegrity theory. The word tensegrity comes from tensional integrity, which is a very Buckminster Fuller term, and he invented it. I don't think he invented tensegrities, but he invented the word.

And they are things like this-- here's a tensegrity built and given to me by Bob Connolly. And it has well, I guess, there are three kinds of edges physically. There are the springs. There are these struts, and there are these wire cables. And its rigid. It's a little flexible, just because the springs aren't perfectly stiff, like all springs.

But there are three kinds of edges here, and we're going to model the three kinds of edges. The edges that we've modeled so far are most closely related to these springs, so that's not quite the right physical analogy. What I would call the edges and linkages is bars.

Bars are, like, solid attachments, like steel rods that are permanently attached to the two endpoints. The lengths of these edges should never change. That's been the world we live in, at least to the first order, whatever. Edge lengths of bars should never change.

Now we have these cables. Now cables really can get shorter. This cable is perfectly happy to get shorter. I can pull those two vertices together.

But what it prevents is getting longer. If I pull-- you can't tell, but I'm pulling really hard on these vertices apart. Nothing happens, because this cable can't get any longer.

So cables prevent extension, but they do not prevent compression. Struts, on the other hand-- they prevent compression. If I push on these two vertices, they don't get any closer. If I pull, it comes right out.

And you have to realize that these are eye hooks, and so really, those could come apart. All these struts are preventing is compression. They don't prevent expansion. So that's the thing we want to model.

This is a tensegrity. It's a generalization of a linkage and where we allow three kinds of edges. It can be a bar-- and this is the old case, which is that the length is fixed.

We can have cables, which is the length can decrease, but it can't increase. And there are struts, which are the reverse-- you can increase but not decrease. Seems like a useful generalization, especially because people build things they actually use struts and cables. Like you look at most bridges-- tons of cables around. I think I have an example here.

There's this guy, Ken Snelson, who makes a whole bunch of cool tensegrity sculptures. This is a particularly tall one in Washington, DC that I saw a couple years ago. That's to give you an idea of height.

And there's this really cool view if you stand-- here's a person. You stand in the center, and you look up. It has this really cool, sixfold rotational reflectional symmetry. I could stare at this for hours. I think that's just the [? zoom ?].

So we want to understand those structures. This one's also rigid, hopefully. It's been standing for 40 years.

So what can we define in this generalized setting of tensegrity? We want motions. We want rigidity. We want generic rigidity. We want infinitesimal rigidity.

All the good things we've defined, hopefully, carry over to this. Well, all but one of them carry over to this.

OK, we can define a configuration space. It's going to be a little smaller, or a little different from our old configuration space. It's a space of all possible configurations where these three things hold-- give you desired lengths for each of these things. As long as these constraints hold, that the lengths stay the same here, get smaller here or the same, or stay the same or get larger here-- that's a valid configuration.

So it's actually a slightly larger configuration space than what we had. A path through that space is going to be a motion, just like before. Configuration is just a bunch of points where this holds-- blah, blah, blah.

The problem is, there's no notion of generic rigidity. For example, this is a generically flexible linkage. We all know it. Now I take my yellow chalk and add struts. Maybe I'll put some arrows on it to make it more obvious.

These are things they can expand, but cannot contract. Then this configuration is rigid or flexible? Flexible, yeah-- this is a one degree of freedom linkage.

The white part is a one degree of freedom linkage, and if you move this thing in a particular direction, indeed, this increases and that increases. Check it.

This one is rigid. The only way to move the square-- or in fact, any generic convex configuration-- is one of these pairs has to get shorter. The other one will get longer, but one of them will get shorter.

So this is the same graph, and the same labeling of which edges are bars and which are struts, but in one configuration it's flexible. The other it's rigid. So we lose generic rigidity.

But infinitesimal rigidity still works, and generic configurations are still meaningful. It's just that they're not all the same. There's not, in this case, for four vertices, there's two kinds of generic configurations. So it makes life a little harder.

But let's say this part will work well. So why don't I be clever, erase this part. So this is the idea of an infinitesimal motion. of a tensegrity.

Well, I want to take the same dot product, and I claim this is really measuring-- it's a number, and it's measuring the sign's change in length of that edge. So if C of v minus C of w dot product with d of v minus d of w is greater than or equal to 0, that's the property I want for struts. That says it gets longer or stays the same to the first order.

Now of course, in reality it might get slightly shorter, but not in the first derivative.

And it should be less than or equal to 0 for cables, and should be equal to 0 for bars. So that's the new notion of infinitesimal motion.

Now, this is no longer a linear system of linear equations. I can no longer write this as a matrix times a vector equals 0. But this is this more general thing called a linear program, where I have linear equations. I also have linear inequalities.

So in fact, I can write it as some other matrix R prime. Actually, it's the same matrix. R prime times d is greater than or equal to 0. This is the general form of a linear program.

And you can write an equality constraint in this world just by taking it and its negation, setting both of them greater or equal to 0. That forces them to be equal to zero. All you need to know is it there are fast algorithms to solve this, also.

So if I give you a tensegrity in polynomial time, you can tell whether there is a valid infinitesimal motion that does all things you want on the struts, cables, and bars. Cool, questions?

STUDENT: [INAUDIBLE]

PROFESSOR: This R-- this is some matrix, and this is your vector of all the d's.

STUDENT: Right. And saying that it's greater than or equal to 0 [INAUDIBLE].

PROFESSOR: Oh, that means element lines, sorry. You take a matrix times a vector. You get a big vector. This is a vector of zeroes. I should have said that. And it's just saying every element in the vector is greater than or equal to zero. So it's just a fancy way of rewriting this stuff again.

But this is what we call linear programs, and that's what's solvable.

STUDENT: And that's just for struts, then?

PROFESSOR: Well, for cables you could just negate the left-hand side. And then that's greater than or equal to zero. For bars you have to include the same left-hand side and its negation. Yeah, I'm skipping some steps. That's why I had to write R prime-- it changes a little bit.

How many people know about linear programming? OK, about 2/3, so for you guys, you also know that there's this thing called linear programming duality. If you have a linear program, the first thing everyone should do when they have a linear program is take the dual and see what you get. I'm just going to tell you what you get. But that's the next trick I want to pull.

So dual notion-- essentially what prevents you from solving that system-- in other words, what makes something rigid-- is something called equilibrium stress. So stress, for me, is just a real number assigned to every edge. And here I'm going to use the word edge to mean bar, strut, or cable.

So a real number s of e for each edge e, and have to have three properties. If I have a cable, then that number should be non-negative. If I have a strut, it should be non-positive. If I have a bar, I don't care-- could be positive, or negative, or zero.

And then there's the equilibrium condition, which is that if I take the sum over all vertices w, where v w is an edge, the weighted vector C of v minus C of w-- so I forgot to mention we're assuming we have some configuration of our tensegrity. This is the zero vector for all vertices v. Let me draw a picture.

I'll draw a real example, how's that? Here's a rigid linkage. I'm not going to say yet which edges are struts and cables. In some sense, I can do that later.

And I'm going to put numbers 1 along all the outside edges and the numbers 3 along all the inside edges-- sorry, negative 1 and positive 3. And then if you work out what this is saying, I'm taking the vectors-- so I'm looking at each vertex like this one, and saying OK, well, I take all of the incoming, whatever.

Let's think of all the edges incident to that vertex. I weight them by this number. In this case, they're all multiplied by 3.

So I take this vector. I multiply it by 3. It gets three times as long. Take this vector, multiply by 3, this vector multiplied by 3.

I add them up. I should get the zero vector. And if this is really an equilateral triangle, I will get the zero vector, so that's good.

And I'm going to draw arrows here to signify what this means intuitively is that this edge is pushing with a really big arrow. This is an arrowhead of size 3. Try to draw it to scale.

Then, of course, this vertex will not move subject to those forces. These forces balance out. On the other hand, this guy, I have these edges of weight negative 1. That means it's going to pull, and it's going to pull with the arrowhead of size 1.

OK, and I claim also that these vertices do not move subject to those three forces. How do you check that? The proper way to check it is to draw what's called a force polygon. Those who have done structural engineering know this.

So I take, let's see, there's this vector, which is length 3 times that, so it would be something like that long. And let's say I'm pushing in that direction, so I go there. Then I follow this thing with the weight of 1, but in sort of the opposite direction, so it will be like this. This is my attempt at copying that edge length and getting what I want.

And then the other force is this one. It doesn't actually matter which order I do these in. And if I end up back where I started, that means the total force is 0 as a vector, and so this vertex is stationary. That's the force polygon.

So you can just check this. It's a geometric condition, if you will, that each vertex is stationary. That is the notion of being in equilibrium. And this is just the algebra to write that down.

Now what the heck does this thing mean intuitively? If your tensegrity is going to be rigid, somehow the edges can prevent a motion. And intuitively, what that means-- I'm going to say this right-- so this is like an edge pushing really hard away from itself, against the vertices.

That's like a strut. A strut prevents compression. So that the force that it can actuate effectively is an outward force.

Whereas cables prevent extension, so they can sort of pull back, say, no, you can't get any longer. I'm going to pull towards me. So this is like a cable. This is like a strut.

Now bars can do both. That's why there's no constraint on bars. But assuming I've got this right-- and I didn't. I got it backwards. Minus 3 plus 1-- it's symmetric so it doesn't matter in some sense, but the intuition matches when cables can pull, and struts can push, and bars can do both, because they prevent motion in both directions.

All right, now, it would be nice to say you're rigid if and only if you have an equilibrium stress. But that's not true. Let me tell you what is true.

So this is duality. This goes back to Roth and Whiteley, 1981-- good year. So let's see, here's what's true-- if I look at all the equilibrium stresses, and I find some stress that is non-zero on a particular structure or cable-- I should mention there's always a trivial stress, which is you set all of these S's to zero. Then everything will hold.

So what we're interested in are nontrivial stresses, where you're nonzero somewhere, and the interesting case is when you're nonzero on some strut or cable, let's say. So if there is such an equilibrium stress, then it's not quite the case that you're rigid, but you're rigid on that edge. Every infinitesimal motion holds the length of that edge fixed.

So this is a strut or a cable. It could get longer or it could get smaller, depending which case we're in. But if there's a stress that's nonzero on that edge, then in fact, it can't change length at all. It effectively acts like a bar in this configuration.

That's what this theorem says. And these are equivalent. If you want to know can I possibly make the strut any longer? You just see can I find an equilibrium stress for that thing that's non-zero?

Another thing you can say-- almost equivalent to this or a consequence of that-- if I have some tensegrity, and I want to know is it infinitesimally rigid, that is the case if and only if two conditions hold. One is that every strut or cable is nonzero in some stress-- in some equilibrium stress-- and I'll say the corresponding linkage is rigid.

So all this is saying is if you can find a stress that is nonzero on every particular strut or cable, then all of those things are effectively bars. So you can convert your tensegrity into a linkage where all the edges are bars. And then it's just a matter of testing rigidity or infinitesimal rigidity of that linkage, which is something we already know how to do.

So in some sense, we already know how to do all of this, because this is just a linear program. We can solve it. But this is a more intuitive way of proving this.

If you want to prove to someone that tensegrity is infinitesimally rigid, you give them a bunch of stresses, one for each strut or cable, or maybe you just need one stress to hit all of them. And then you show that the linkage is infinitesimally rigid, which you could do by computing the rank of the rigidity matrix. Strut can have zero stress-- the weight is zero.

OK, let's keep going. Lots of a cool things to talk about, maybe over here. I want to do a little diversion for fun, something called spider webs, like this. You might have seen them before.

A spiderweb is something where every edge is basically a cable. You can see under wind and tension, they get a little bit shorter than they're supposed to be. But they can't really get longer. They are pretty strong at preventing expansion.

So a spiderweb is something that can be stressed with all positive weights. And the reason it holds in tension-- positive, yeah, cables-- all of the edges can have a positive stress, and it will be in equilibrium. And in spiderwebs, in fact, this is the same-- these two-- rigidity and positive stresses are the same thing.

This is sort of a special case, and Connolly studied these in 1982. There's some special points in a spiderweb. Here they're kind of off the page.

There's a point down here where it attaches to something, and up there, and up there. Those guys do not satisfy equilibrium, or you can say those edges have some negative stresses. Or another way to think of it is there's a point off to infinity, which is how this is drawn, and you can have a positive stress. And at infinity, you don't have to satisfy equilibrium.

So there's a little something special at the boundary, but everywhere else-- interior and all these vertices in here-- you have equilibrium, and all the stresses are positive. And it turns out in this case, you're automatically infinitesimally rigid. That tensegrity is infinitesimally rigid. Normally, we need to also check the underlying linkage is rigid, but for spiderwebs, you don't have to.

Why do I care about spiderwebs? Because they give us something called origami tesselations. This is a brand-new result from this summer at [? OSMI ?] by Robert Lang and Alex Bateman. Here's a little example of an origami tessellation, this fold by Jenny [? Ramsier ?], just the one I had hanging around in my office. It looks a little cooler that way.

Let me hold it up to the light. I can see it great. You don't see the light. It looks pretty cool.

It's folded from one square, and it's folded in some sense with a regular pattern. You can see regular hexagons, hexagonal pattern. And these guys have been around for a long time, probably at least the '60s, maybe earlier-- a lot of Japanese artists, lot of American artists. And there's been a lot of excitement about origami tesselations since the mid '90s.

I think Chris Palmer revitalized them, and then tons of people are looking at them. And there's this cool algorithm for building them.

So you take some tessellation like these squares, and triangles, and hexagons-- some collection of polygons. You shrink all of them a little bit, but all by the same amount. And then you twist them a little bit, all the same amount.

So here they are twisted. And then you just connect the dots. So these two edges used to be joined to each other. So I'm going to connect these guys, and connect these guys. That's the green edges.

And that's your crease pattern. You erase the red stuff, and you get your crease pattern, and you fold it, and you get this cool tessellation-- magic. Now the question is, when does this work?

This algorithm has been around forever, as far as I know. I don't know exactly how old it is, or who invented it, but it's pretty natural. It's how most tesselations are folded.

But when does it work? And there is a brand-new result from the summer that it works for some choice of those angles. So I had some flexibility-- how much do I shrink and how much do I turn? But this will work for some choice of turn and shrink amount, if and only if your graph or your initial tessellation is a spiderweb.

Pretty cool-- spiderwebs have this neat property that you can do the shrink and twist tessellation algorithm, and it works. And here is an example of that-- a spiderweb, so to speak, but drawn visually. And this is actually a proof that-- these are basically all the force polygons showing that you can get a positive equilibrium stress.

And you turn it into a crease pattern, fold it flat, and it works. This is, of course, in simulation, but you can really fold these things, and they work. So that's kind of fun-- a little bit origami out of linkages and tensegrities. Questions?

All right, then we move on. I want to tell you one more cool thing about stresses and tensegrities-- yet another characterization of stresses, which is polyhedral liftings.

A polyhedral lifting is going to work if I have a non-crossing configuration. So I don't want to have this-- two edges that cross each other. I only want to have pictures like this, where edges don't cross each other. They only share endpoints.

Then I can define what's called a polyhedral lifting. So this is a planograph, if you will. Polyhedral lifting is going to be an assignment of z-coordinates to the vertices. So it's going to be a z-coordinate z of v for each vertex v such that each face remains planar.

Because I forbid crossings, I end up decomposing space-- the plane here-- this only works in two dimensions. This is one thing that does not generalize to arbitrary dimension. So I get a bunch of triangles here.

And so I want to lift. I want to assign some z-coordinate to each of these vertices, such that each of these triangles remains planar. Well, actually, that always holds. Triangles are always planar.

But there's another face, which is the outside face out here. That's a quadrilateral. That one also has to remain planar. So these four vertices have to be lifted onto some plane.

This guy can be lifted however it wants in this particular picture. In general, you're going to have a lot of non-triangles. Those should all remain planar. That's a lifting.

One thing you can do-- there's some freedom here. If you have any lifting or, for example, you could not lift them at all. Set all the z-coordinates to zero. That's fine. You could also just lift everything onto some other plane, and generally have a rigid motion of freedom.

You can't translate around, but you have one two-degree two-rotational freedoms, I think. And so we would just like to get rid of those. So say remove rigid motions by forcing the outside face to lie in z equals 0.

So let's say these guys all stay at 0. So in this particular example, this guy can go to any value, positive or negative, but that's it. Now, this is a lot like stresses-- in fact, it's identical to stresses.

It's probably not obvious, but just like stresses where you could set everything to 0, here you can also set everything to 0. It's not a big connection, but it turns out they really are the same thing for non-crossing configurations. This is called Maxwell-Cremona Theorem.

This is probably one of the oldest theorems we will cover here-- I guess anywhere. Well, it's not that old-- 1864. That might be the oldest. Maxwell proved this, or claimed it in 1864.

Theorem is there's a one-to-one correspondence between the equilibrium stresses and polyhedral liftings.

And fun fact-- negative stresses correspond to valleys. So valley is something like this. You know what valleys are, but it's maybe a little bit subtle. We're not folding all the way here, so this would also be considered a valley, because it has sort of negative dihedral triangle.

It's really hard to see. What I meant was something like this. That's a valley edge here. This is another valley edge here-- whatever.

A positive stress corresponds to a mountain, which is the same picture upside down. A zero stress corresponds to a flat angle-- doesn't have to be horizontal, but it has to be flat. So it could be like this.

I'm going to need this in a moment proving the next thing. I'm not going to prove this theorem, but it would be fun to actually see it in action. I've never seen it implemented, so another fun project.

I give you tensegrity. You compute an equilibrium stress, which is easy to do by linear programming, and draw-- you can directly construct from that the 3D lifting, which would be kind of cool. I'm going to use basically everything I proved today to prove a big theorem-- the Carpenter's Rule Theorem. It's why I'm wearing this t-shirt that has nothing to do with the Carpenter's Rule Theorem.

And it's about non-crossing linkages. So this is the beginning. So far we've been allowing crossings, except for this very last theorem.

Now we begin this section where I forbid crossings between the edges, just like what that picture was. Prevent this, and this is a constraint on the configuration space now. I'm going to look at all the configurations where no two edges cross.

This is a smaller version of our old configuration space. I'm not going to define it formally. It's a bit messy. It's not too nasty. It's still defined by polynomial inequalities. Question?

STUDENT: What if you have one of those diagonals and switch it to be in relation to each other [INAUDIBLE]?

PROFESSOR: You want this picture and--

STUDENT: Either one.

PROFESSOR: And you want to add this? Or-- that's crossing-- not allowed. Whatever you're saying is not allowed. That's the easy answer.

I don't quite know what it is, but what we allow-- and in this case I'm just thinking about linkages for the moment. It's just you can hinge at common vertices. Two edges either share a vertex or not. And that's it. But they're not allowed to touch anywhere else.

And now what we care about is something called a locked linkage. And a linkage is locked if its configuration space is disconnected. So the configuration space would look something like this. Maybe it has three components.

If I have some configuration here, I can move to any other configuration here, and same over here, but there's no way to get from this configuration to that configuration. They're disconnected from each other. So this would be kind of sad news if you're looking for motions. It means you cannot get from everywhere to everywhere.

There are some locked configurations that cannot get back to start. However you define start and locked doesn't matter, but the linkage is locked when there's two things that cannot reach each other. And the Carpenter's Rule Theorem is about one case of that.

So this was essentially my Ph.D. thesis, so paper in 2000 by Bob Connolly who made that tensegrity, [? Gunta ?] [? Rota ?] and me. And it says that if we have a linkage, and say we take a configuration of that linkage, and let's say it's of maximum degree 2, so every vertex has at most two edges coming out of it. Then there's a motion of that linkage that straightens out-- so if you have maximum degree 2, you have paths, and you might have some cycles. You might have many of them.

And it's going to straighten all the paths and convexify all the cycles. And it's also expansive. But before I get to that, this is essentially showing that these configuration's bases are connected. It doesn't quite imply it the way I've worded it.

But for example, for paths, I take some path. I want to know, can I get from this configuration to some other configuration? Well, I could take this one, find a motion that straightens the chain. I could take this one, find a motion that straightens the chain.

So here I've drawn it straight. I take this motion. I apply it forwards. I take this motion. I apply it backwards, and I get from here to here.

So if I can straighten all the paths, I can definitely get from anywhere to anywhere. For convex cycles, it's a little less obvious, but it's also true. There's one catch which I didn't say here.

I need to add outermost. When you have nesting like this, you're in trouble. This guy is not going to get straightened out.

It could be super long. It may not have room to straighten out inside that convex chamber. So these guys will come along for the ride, but they won't actually get straightened. The outermost guys will get straightened and convexified.

The other fun property-- and this is how we actually proved the theorem-- so this was open for a long time. People thought this was true. And the key to proving it is by making the theorem stronger, which is that all pairwise distances between vertices increase or stay the same.

Some of them have to stay the same, because they're edges. They are bars in there. They can't change in length. But everything else will increase.

Now there's a couple reasons why this is useful. One is that it says you don't have to worry about crossings. If I have some edges that initially don't cross, and I'm worried about them crossing, that can't happen if you're expansive.

If I say, well, this distance has to increase, and this distance has to increase, there's no way that this vertex is going to cross over that edge. One of them would have to get shorter. So this essentially says, everything's flying away from each other.

So if initially you're non-crossing, you will be non-crossing forever. That really makes life easier, because crossings are hard to think about. This lets you not worry about crossings. You just have to worry about distances.

How do I worry about distances increasing? Tensegrities-- struts-- struts will force things to be expansive. If I want to model this picture, all I need to do is add a whole of struts. Let's go over here.

So we're going to prove this theorem in, like, five minutes-- easy. Once you build all this technology, which was existing, you can prove this theorem, once you realize this is the theorem you want to prove-- this expansiveness.

So what we're going to do is build a tensegrity. And it's going to have a bar for every edge of the given linkage. So that's just the same thing as we had given.

It's also going to have a strut between every other pair of vertices-- heck, every pair of vertices. It's kind of redundant for the bars, but every pair of vertices. That says that all pairwise distances must increase or stay the same.

So this is a tensegrity. What I want to prove is that this tensegrity is flexible. If I can prove that it's always flexible-- well, until the end, which is when everybody's straight or convex-- all the outermost guys are straight or convex-- then I'll have proved this theorem.

So while this theorem is about actual connectivity of configuration spaces, it says paths and cycles have connected configuration spaces. You get from everywhere to everywhere. It even gives you an algorithm for doing that, which we'll talk about next class.

We use just, is this thing rigid or flexible at all? And that's the power of all this rigidity stuff, just telling whether something can move a little bit, and then just doing that over and over again, you will be able to find that you can actually go all the way, and get to straight or convex. But how do we actually prove this thing?

So the point is, claim this tensegrity is infinitesimally flexible, meaning it is not infinitesimally rigid. And that will imply that there's at least an infinitesimal expansive motion. And then you have to use some fancy tricks-- not fancy, some tricks I don't want to talk about. You basically integrate that vector field, and that will give you an actual expansive motion.

Let's not worry about that. The interesting part is show that at least infinitesimally, this tensegrity moves a little bit. So what do know? Duality-- if I want to show not that it's rigid-- I want to show that it's flexible. But it's rigid if and only if this is true and this is true.

This will actually always be true-- the corresponding linkage is rigid-- because I put in every edge is in there. If I turn them all into bars, this thing's not moving at all. So fact, this second condition doesn't really matter.

What matters is the stresses. If I want to show that it's flexible, not rigid. I want to show these stresses don't exist. Now stresses always exist. I could set all the stresses to be 0.

But I claim actually pretty much all the stresses have to be 0, at least on the cables and the struts, which is what I care about here. They have to be 0. So this will be implied by every equilibrium stress is 0 on all struts and cables.

So you can never get this thing to hold for any of the struts and the cables. And therefore, in some sense, none of them are fixed, and therefore, you're actually flexible. That's implied by this duality statement.

OK, stresses are a little hard to think about. I prefer to think about polyhedral liftings, let's say. OK, the key here is to plug together all the right results. This will be true if and only if every polyhedron lifting is flat.

So I guess in our case, it will be 0, because the outside face sets all the z-coordinates to 0, so everything else will have to be in that plane. So all the z-coordinates will be 0.

So these are equivalent statements. All I need to prove is that every polyhedral lifting is flat. So we're going to prove this theorem by contradiction.

Suppose there was a non-flat polyhedron lifting. What could it look like? Did I just erase what I wanted? Probably.

Greater than or equal to 0 for cables, less than or equal to 0 for struts. That's the one I care about. Who cares about cables?

All right, so this is the stress. And equilibrium stress has to be less than or equal to 0 for every strut. Less than or equal to 0 corresponds to a valley edge or flat edge. Now in this linkage, almost all the edges are struts. There's a few that are bars, this, like, little path is a bar.

Those could be valleys or mountains. We don't know. But everything else has to be a valley or flat. For flat, we're good. We're happy. We want all the edges to be flat.

So how could it be that almost every edge in this picture is a valley? That seems a little tricky. And we're going to show it can't happen. If everything's a valley, something's going to go wrong at the top picture, because you've got to have mountains, like, turn around.

So let's look at the top. Look at the maximum z-coordinate. Now this could actually look like many different things, but I'm going to consider one case-- the easy case-- where you have, in the maximum z-coordinate, you have one point. And then around that, everything descends.

OK, these are all the edges incident to that point. You form some kind of-- OK, so this is maximum z. This is another plane. I'm going to slice that, which is maximum z minus epsilon-- slightly below the maximum z-coordinate.

And in a sort of typical case, we're going to get a polygon. Now note that this polygon has this nice property, that wherever it has a convex angle, you get a mountain here. Wherever you have a reflex angle, like this one, you get a valley edge here. You can see that 3-dimensional picture.

Every polygon has at least three convex vertices. That means there are at least three mountains incident to this maximum z-coordinate But mountains have to correspond to positive stresses. Positive stresses have to correspond to bars, because struts can't carry positive stresses.

So that means you have at least three bars incident to that edge. But I assumed I was maximum degree 2-- contradiction. Isn't that cool? It's very easy when you set it up right.

This works almost always, but there is another situation which can happen, which is, like, you have-- in the maximum z-coordinate you might have a couple of bars at the top. That's a worry. This is all in maximum z.

And so when you slice below, you don't actually get a whole polygon. You might actually only get something around here. But again, you can't get from here to around there without some mountain.

It's hard to even imagine, because it can't happen. But this is a mountain, and this is a mountain, and you want to somehow slope a surface below this plane, and get around to over here with only valleys-- ain't happening. You've got to have another mountain in here. And so that's the general picture.

And the only thing that can happen is you actually prove that all of this stuff has to be flat. So that's the one way it can happen. You can't have strict valleys, but you could have them all 0.

So in fact, the one case where you can have stress is when you have all of this outside stuff flat. And then inside you don't know. I can prove, using this generalized lemma, that all of these regions are locally flat. And therefore, this whole outside face has to be flat.

Therefore, I'm particular. I can't have a reflex vertex, because then this side would have to all be flattened. So then the whole thing is flattened. The only way the whole thing can't be flat is if your polygon is already convex, and then we're done.

Also straight this can happen. I think with straight, you don't get any stress. Here you can actually have a lifting. Actually, all these guys would be below the plane, because everything's a valley.

This would be like an actual valley, where you have a village or whatever. That can happen, but only when the boundary of your valley is convex. And so that proves the theorem. And that is the Carpenter's Rule Theorem.

Next time, we're going to talk about the other situations. This was two dimensions, and maximum degree 2. We can think about what happens with degree 3. Then you can get locked things.

What happens in three dimensions? Then you can get locked things. What happens in four dimensions? Then you can't get locked things, and other fun things like that.

Oh, and why is it called the Carpenter's Rule Theorem? Because this is a carpenter's rule. And in a carpenter's rule, actually, all the edge lengths are the same. But as far as we know that doesn't make this theorem any easier to prove.

And also in the carpenter's rule, you can have crossings, but other than that, it's just like a carpenter's rule. So there you go. So if you've never played with one, this is what people used before flexible measuring tape. They still use them a lot in Europe, but not so much in the US, but you can still buy them at the hardware store.

That's it.