

**ANDREW**

Good morning, everyone. I guess it's 8:00 now. So I think we'll probably look at getting a start.

**MILFORD:**

So this is a really good turnout. This is my first time speaking at AU, so it's good to see such a large number of people on a Thursday morning.

My name is Andrew Milford. I'm an AEC Technical Consultant for Autodesk Australia and New Zealand. I've only been with Autodesk for about 18 months. Prior to that I've spent about 25 years in industry, working for companies such as AECOM, Arcadis, and Jacobs Consulting. So my background is heavy infrastructure, road design, geometric design, major roads, highways, interchanges.

I also contribute to a blog, "From the Ground Up." Has anyone actually visited this in the past? Yep. Junction Jack Strongitharm was kind enough to invite me along to write about this. So I'd like to give it a distinct Aussie flavor. I like to call it "From the Ground Downunder" just to localize it for the Australian market.

So we'll get into the session. We've actually a lot of material to cover today. If you downloaded the notes, the presentation notes, you'll see that there is actually like a 55, 56 page document. That's actually a much more comprehensive list than what we're going to go through here today since we've only got 90 minutes to condense it.

So what we're going to run through today, this exercise is going to split into four parts, four exercises. Each one is going to be split between Subassembly Composer, where we're going to introduce new concepts. And then we're going to apply those in the Civil 3D environment in a real world scenario.

The first example is going to be exit ramps. This is going to be an introduction to subassemblies. We're going to take the basic four layer pavement generic subassembly that ships with Civil, and we're just going to extend its functionality. We're also going to look at creating what I like to call a slow projection, where we can start to manipulate our crossfalls and match slopes from existing surfaces.

Exercise 2, we're going to build upon what we learn in Exercise 1. We're going to take and look at the concept of transitioning across our regions in Civil 3D. Now, when I first moved to Civil 3D, I found this really difficult to do, getting that whole concept of batter transitions. We're also going to look at the concept of linked corridors. So we can actually break our model up

into a much more manageable components but still keeping the corridors linker in the event of change.

Exercise 3, pram ramps and kerb laybacks, we're going to build upon what we picked up in Exercise 2 with the transitioning formula. And we're going to actually apply it to a real world pram ramp and kerb layback scenario.

Traditionally I see these things modeled up using feature lines in Civil 3D. I'm pretty lazy. I don't like to actually do that. I like things that are dynamic and are going to stay linked back to my parent corridor. So we're just going to build this and actually use it as part of our corridor.

And finally, the behemoth, we're going to bring it all together and create some complex batter slope transitions. We're going to use the concept of linked corridors, auxiliary alignments, offset alignments, to build out a multi-bench batter slope configuration that you're going to have complete control over and be able to edit and manipulate very easily.

So if there are any questions we'll hold them off until the end. Because we've got a lot of material to cover. So that's my standing up part. I'm now going to sit down and start to drive. I don't like PowerPoint. Oh yeah, at the end of this class-- you guys have all read that. Haven't you?

So we're just going to really run through applying the alternate modeling techniques, what I like to call a linked corridor technique. We're going to create some complex transitioning. And we're going to explain some of the [? mattes ?] behind it and how we can actually use it, not only for things like batter slopes, but for any kind of component within subassemblies.

We're going to look at how we can create, edit, and reuse these using Subassembly Composer. Is there anyone here using the Subassembly Composer? For those that aren't, definitely check it out. I've actually included all of the packet files in today's demonstration on the website.

Download them. Pull them apart. Play with them. Make them better. They're not perfect by all means, but they do work. But you can certainly take them and make them your own.

And, yeah, to me it's all about reducing the rework time. Maybe a little bit of upfront cost in the set up, but downstream when the changes occur, and as civil engineers, they change all the time. We have models that can actually respond to that change very quickly.

So the first exercise we're going to look at an exit ramp scenario. And we're going to look at modeling this up just using first principles techniques. The difference between this exit ramp-- and this is just a shot of an exit ramp that we use in Australia.

You can see here we've got some tapers down here. This is the one we're going to be focusing on. We've got our gore area. It shoots off at a 1 in 15 taper. Gets a little complex. But what actually is going to make it more complex in Civil is the exercise we're going to do-- we're actually going to do it on the outside of a curve.

So we're going to be developing super elevation in between this area here. So how do we actually create multiple corridors that can actually talk to one another and still maintain that relationship with the slope?

So to do that, we're going to-- this is our first of our two subassemblies. This is the six layer pavement. All I'm doing in this exercise is taking the four layer pavement that ships with Civil 3D. I'm adding an extra two layers because we use-- I don't know. We use quite a few more layers in Australia. We use slick material, base, sub base, plus multiple pavement layers. And we're going to add some marked point functionality, so we can actually work from the outside in, not just from the base line out.

The second is a really, really simple one. But it's going to show some of the real power of Subassembly Composer. This is called a-- I just call it a slope projection. All it's going to do is take two points on the surface and return a value, which we're going to use via parameter referencing to control the slope of our tilting roadway.

So I'm just now going to jump into Civil-- no, Subassembly first. And this is actually, at the moment, five lanes. If you haven't actually played with Subassembly Composer before, it's really just a case of grabbing your points, links, and shapes, dragging them across, and editing them from here.

What I like to do is always like to put a version output parameter here. And I do this in every single one of my subassemblies. So I've got version control. So when I ship it out and push it out to users in the teams I'm working with, they can always know if I've got the latest version or not. And make sure, most importantly, everyone is on the same version.

We're going to use a-- this is called a defined variable. We're actually taking the lane slope calc here. At the moment it's set to a value, down here as you can see, of minus three. We're

going to plug a formula in here. You see on the list here, we have all the input and output parameters. Now, I've actually enabled control for point codes and link code. So the users can actually enter their own point codes as opposed to just using the stuff straight out of the box.

The lane slope calc-- I'm going to cheat a little bit here and just grab a formula. Because it's actually quite long, But quite simple when you break it down. So instead of minus three, we're going to take this. You just extend this. And all this is saying is, if this parameter here, called U super elevation, is equal to the left inside a left outside lane, and it actually has super on it, use that value.

This is just using Subassembly API at the moment. If it doesn't actually, if that does not conform, if it's no, then go down to this line, and so forth, and so forth, and so forth.

It's set up just like you doing in Excel if-then statements. The only real key here is, when you get to the end, saying if there is no super elevation, use this default slope value. That's all it's doing. And our default slope value is a parameter sitting down here in our input outputs.

Don't be frightened off by the C type double. All that's saying is, the super elevation value is a double. The default slope, as you can see over here, is a grade. All we're doing is converting that type of a grade into a double.

So when you're actually doing these formulas, you need to make sure that both the true and the false condition statements are of the same type. In this case they're going to be a double value.

So once we run that in, we can feed it into the top layer here. What I'm going to do in this case is-- this is how it originally ships out of the box. I've already set up a condition statement saying, do I want to use a marked point?

That is actually driven by this yes-no parameter. So if I'm going to use a marked point, I'm going to actually re-hook-- that's a nice one to see. And just undo that. I'll take these links out. Move that across. And connect into that.

So we're calculating the super. Are we using a marked point? No we're now. We're just going to use a standard-- let's just move that out there. We're just going to use a standard super elevation value. Then all we're going to do is connect it back in to create the geometry down here.

If we do want to use a marked point, I come over here, and I use this marked point instead. What I like to do when I'm designing these is I love to actually just either sketch them out by hand first, either on a piece of paper or in CAD so I know what parameters I'm going to be dealing with.

The Subassembly Composer does not let you move these up and down. That's the problem. So it's nice to have them all in sequence first so it's just nice, and neat, and clean for your users at the end. So you can see here, I've got an error warning. It's telling me that my link isn't working so I need to make sure I'm hooking up between P-1 and P-2, error goes away.

We connect back in here. And then we just continue down the chain. So we're just working in a very sequential order here. So pavement five, what we're going to do is now-- this is how easy it is to use. We copy and paste all of those elements. And we're going to make a sixth pavement layer.

All we're doing here in this decision is saying, if the pavement depth is equal to zero, don't give me any thickness if it is. Give me pavement depth. So we run those in through there.

And you can see we've already got the sixth pavement. But the one thing we haven't got here is-- I just need to make that pavement six. Is we just need to change the points they're linking off. So I P9 goes to P11 So you get this interactive feedback here on the side. P10 becomes-- we want that off P12. And we want to move that link, you can see there's a bit of overlap there-- and we just want to run that between P15 and P16.

So there's our sixth, sixth layer. So we can build up as many of these as we like. 10, 15 layers, if you want to put in your binding layers, which I've seen people do. I don't know why, but they have.

And then to create the shape, we just select, come in here. subassembly Composer is pretty intuitive. I could just start typing in link values between L20, L24, 23, 22. I'm just going to click Add Link a few times. And it picks it up. So we've now driven the shape within there. So that's really all there is to that component of the subassembly.

The next one I wanted to show-- and this one is really, really simple-- is the Slope Projection Tool. This doesn't actually return any geometry. It's just going to return a value. That value then we feed in, through parameter referencing, into future subassemblies when we line them up in our final assembly.

So all we're doing is putting two auxiliary points up here on the left-- on the right, sorry. I'm just going to link that in to there. Now, within these two sample points, AP1, AP2, all we need to do is a simple calculation to work out what's the slope between the two that we can then feed downstream to control our geometry?

So to do that, I'm going to use a define a variable. I'm going to call this existing slope. So we can give the variables a name. I'm going to call it Type Double.

And the formula here, this is actually using the Subassembly Compose API. the Help on it is really comprehensive. It's really good. So hit F1, by all means, and just start looking at it. It's AP1 dot slope 2, open bracket, quote, AP2.

Let's get that. That should, theoretically, go away. Nope. That formula there is just going to go and get the slope between those two points.

And when you're actually driving that slope, these two AP1 points are actually sitting on target surface. So when we set it up, we need to assign the target surface. It'll just go and grab those points for us.

And I'm going to show you how we can now apply that. Better hook up my point here. SA.IsLayout, this is just another quick one. And if that equals true, SA.IsLayout is really saying, what's it going to look like when I put my subassembly into model space? So I always like to put one of these at the top of most of them. Just so I can control the final look of the subassembly.

If it's false, we're just going to output that value, called Out Lane Slope. It's going to push that value out. And that's really all there is to this one. But it's really easy and really good to use.

So we'll jump into Civil 3D now. Make sure I got the right drawing. And we'll start to look at how we can build out this ramp.

Now this drawing here-- actually I'm going to just pull this across to here. Everyone see that all right? Yeah.

We have our through lane here. It's actually got super elevation applied to it. You can see we've got some super elevation values down here.

I've got my exit ramp control coming out here. And I've already built out a lot of this, because a

lot of it's just basic corridor modeling. The tricky part is through here, where we're going to be rolling out super elevation through a widening area with multiple base lines.

So a few ways we can go about this. I've already built out a temporary planes model here. Does everyone work with auxiliary corridors and temporary planes doing it this way?

The concept behind it is, you build up these temporary planes. This is just two generic links that are using the super elevation values. I'm creating a working surface. And all I'm doing is creating a surface in there that's linked back to the master alignment.

What I'm going to do is drive the geometry of this exit ramp control here by projecting it onto that surface. So this is my MC30 control at the moment. I've got my existing program profile. This is the MC10 line that's being projected onto that view.

Now I'm just going to turn on the surface in here, the temporary plane surface. If I hit apply and OK, you could see, that is our temporary plane surface running through there. So you can see our super elevation is actually rolling between negative 3% and positive 3% through that area.

All I need to do now is actually start designing off one of these points here as I exit the gore area, once I get past the nose. So I'm going to turn on my design control as well, just in the interest of time. So should my alignment move at all, this purple line you see here is going to move and follow it as well. So all I need to do then is just pick out my exit ramp string and just adjust it accordingly.

So back to the corridor model, I'm going to look at-- I've already got a section in here. I'm going to use the MC30 base line to extract levels from the through road super elevation. First though, I'm just going to look at this taper that we're looking at here.

Now I'm not going to use a custom subassembly for that. I'm going to use what we call a number transform. And this was built into ANZ country kit earlier this year in 2016. So coming to our subassemblies, I'm going to use this thing called the Exit Ramp Diverge Taper. This is just going to show some of the flexibility of parameter referencing and how we can get some transitioning on, not just slopes, but pavements as well.

So you can see here, I've got my two through lanes. Let me just drop the scale on that. It's gets a little cluttered. That's my exit ramp pavement. And that's my shoulder.

What I want to do is transition from-- there's my two through lanes with my shoulder. I want to transition from here out to here. So to do that I need to actually insert this thing called a number transform. Now all this is going to do, if I look at the parameter here, it's going to give me value 1, value 2, and a transform type.

I'm going to start at 0 offset. I'm going to finish at 3 1/2. And I'm going to insert it just after that second through lane pavement. Now it looks a little bit odd. But all we need to do is come into our assembly properties. And this number transform, I like to give it a name. So I'm going to call it Widen Value [? zero ?] three and a half linear.

Now I need to make sure that my actual land ramp configuration is going to inherit the values of that transitioning number. So my width, currently set to 1 1/2 meters, I'm going to override it with that output value.

So I'll show you what this does when we actually apply it. So I'm going to select this. I'm going to add the region into the corridor. And I'm just going to put it in that range there. And it's the Exit Ramp Diverge Taper. Now I'm going to put my final existing ground. And I don't actually need to fill in any of these.

So you can see what's happening there. That lane that was 1 1/2 meters wide is actually transitioning from zero over here-- move that across-- up to 3 1/2 meters there. The great thing about this is, because it's a region, I can see that's actually too wide. If I need to change that or move it, I just simply do that. And my taper changes. So we can apply these sorts of ranges to practically any subassembly feature within any subassembly within Civil 3D.

So I'm just going to come into my corridor properties and turn on my second region. So this is actually a-- you can see this one's actually being built out specifically to use line marking as well as solid pavements. So what I need to do now is just add this final piece through here.

If I just come up and show my sections, you can see what's going on through here. I always like my section string. So we're at negative 3% through here at the moment. If I run this through, you can see we're actually starting to develop the super elevation through this area.

Now to try and match MC30's super elevation to that value and get it to line up exactly, it's going to be near impossible. So what I'm going to do is actually use the values of these temporary planes and that slope projection tool to actually drive the slopes through that curve.

So that's what we're using this here. I've already built out the two components. This here is

the-- I'll just [INAUDIBLE] that back. This is the slope projection tool that I mentioned earlier. I haven't actually hooked it up with the parameter referencing. So up In our assembly properties-- that's the parameter referencing, sorry-- that's the slope projection tool that's currently used. At the moment it's sampling two points, one meter, zero, so it's sampling it on the baseline. And the sample point offset one meter away.

I just need the shoulder here. I need its default slope to inherit those properties. So I'm going to take the lane slope of that. And down here on the right hand side, I need to do it again. So, take the width-- sorry, take this slope. And take that slope value from the previous section.

Is anyone using parameter referencing here at the moment? The rule with parameter referencing is you need to have this slope can only inherit values from subassemblies that are higher up in the hierarchy. So if you put it below, it will not pick up those values. And you cannot select them. That's the only real rule to apply there.

So with this, I'm going to add another region into this corridor, which is my MC30. I'm going to use this exit ramp section. I'm just going to use this alignment here. I'm just going to run out-- so you can see there, I've got a little annotation point there. That's what I'm going to use to carry my super elevation on into the exit ramp, once I reach that point.

Exit ramp at gore. Now my batter transition, I'm just going to batter to existing ground. The only thing we need to take care of in here is, obviously I've set up the slope projection. But it needs a surface to drive the levels from. In this case, it's going to use the temp planes model that I've generated underneath. That temp planes model is the one you can see here over on the right, this big, this red line that's following through. So it's going to pick up the levels on the surface there and push through.

And I just need to change that one there. Because I'm picking up another string. Select my layer.

So you can see it's pushed all the geometry through there. I'll add it to the section here. Let me just pop that on. MC30 exit ramp, and I'm going to make sure it's-- I like to annotate it at the actual cut points.

So you can see, just by taking those levels through there, if I come back through here, you can see it's actually projecting through. There's my temporary plane surface. It's actually picking up the levels on that surface and actually driving the geometry through there

So any changes I make to this alignment now, it's going to be fed through and automatically update. So I don't have to bother recalculating my super elevation through that area

Clear as mud? There's detailed descriptions in the document. I'm sorry if I'm going through quickly at the moment.

OK, we'll head back to the slide show. This is just some of the examples of the number transform subassembly that I'll use first. I say it's a really powerful tool. I highly recommend using it. You can get some really cool results out of it. I only use the linear one up at the top there. But as you can see you get parabolic out, parabolic in, reverse parabolics. You can control practically any kind of curvature through those transitioning regions.

So we're going to move into Exercise 2, which are batter slopes and transitions. This is where we're going to take the concept of transitioning. We're going to explore the formula used to actually generate the transitioning code. And what we're going to do is we're going to build up a batter slope configuration that can--

We're going to add some decisions in Subassembly Composer. So we're going to determine first, are we in cut? Or are we in fill? But not only that, we're going to add additional conditions to determine are we in shallow fill or are we in deep fill? And we're going to adjust our day lighting of sub grade accordingly.

It sounds pretty complex. But it's actually not too bad. Typically when I like to model my roadways-- now you can see the transition. You can already starting to see what we can do with it. When you are traditionally modeling a roadway, it's actually rather-- the roadway component itself is the easy part. Yeah, you've only got a handful of regions that you can be using throughout the entire process.

Everything that's complex happens from the hinge point to the batter point. So I don't know how it's done here in America. But in Australia, we do a lot of transitioning. We have two to ones, four to ones, five to ones, depending on the conditions we're in. We could use civil 3D's stock daylighting subassemblies.

We could use targets. But what happens when your alignment moves? Your targets become obsolete. How do you know if you're getting exactly two to one? How do you know if you're getting exactly four to one?

So the subassembly we're going to build out here is actually going to give us complete control of not only the values, but where we can actually place them. And it's going to break the model up into a much more manageable components, and I'll show you how we can do that through the use of a linked corridor.

So this is a quick look at the batter transition subassembly. You can see, we've got the slope running down here. This is a high fill situation. We've got a depth parameter that we're going to be adopting. And we're going to have a slope on the road. And we're going to apply some automatic daylighting of subgrade.

So we're going to dive back into Subassembly Composer for a brief moment. And we're just going to look at some of the formulas.

OK, once again, for the input-output parameters, we've got Slope 1 which is the start transition, and Slope 2, which is the final transition. You can see here. There's not many parameters involved. But this thing does an awful lot in terms of giving you control and flexibility over the transitions.

The slope of the road, that the slope of our subgrade. I could be a bit clever with that. I mean, you could make it actually follow the super elevation of the road just by using a super elevation parameter. I was a bit lazy in this case. It works.

So the first thing we're going to do, we have an AP1, which is our auxiliary point sitting on the surface. And we have this delta slope variable that we're declaring. This value at the moment, we need to get really just the delta between what's the difference point Slope 1 and Slope two? It's just going to form part of the final calculation.

So that's just going to be Slope 2, which is this value over here, minus Slope 1. Once again, we've got the subassembly is layout equals true. Everything here you see cutting off to the left from the decision tree, that's all going to-- if I come to layout mode up here and fit to screen, that's really how it's going to look when I place it into model space. That's all it is.

If the layout is false, we're going to come out here and create-- we're going to change this delta slope value to calculate the actual final transition value at a specific baseline station. So to do that, we've defined the variable. Now we're just going to set that variable value. We're going to basically reset it.

So this variable-- just need to first connect that in. And it's picking up the delta slope. So we're going to change it. So this is the magic formula, as I like to call it. Cause when I found this, it just opened Civil 3D up to me completely. It just completely changed everything.

Double the open bracket, this is the subassembly API. Once again, baseline, being your baseline, dot station. So this is the parent station value. Minus baseline dot region start. I'm not expecting you to remember this. It's all in the documentation. I'm just typing it out to remember myself. Baseline dot region end minus baseline dot region start. Close bracket, close bracket.

So all that's doing is saying, the current baseline station, the distance from the baseline station from the start of the region over the entire length of the region. It's going to give us a percentage between 0 and 1.

And then all we're going to do is multiply that value times the delta slope. OK, so then we feed that into the new point here, Point 1. Need to come up to my roadway mode. So there's Point 1 sitting on my new-- just sitting above AP1. You can see here we have a target surface parameter There is fill. There is cut.

So the first of our decisions in here-- all we're saying at this point here is-- you can see this formula down here, AP1.Y, which is here, basically the zed value. So is AP1.Y greater than this point here at our hinge point? If the answer is true, we go into cut, which is just over this side here, where we're going to daylight to existing surface.

Now you can see here, nothing's happened. The condition is true, but it's not happening. When you're doing these sorts of formulas, you have to actually bypass the calculation just in order to get a better visual look when you're designing this. So I'm just going to bypass it temporarily. And you can see now that cut situation is true.

If I come down here, obviously if it's false, we're going to head into fill. And we're going to head down into this component down here. We're branching off. So you see here, there's the target surface. We've got out depth, our slope, and our auxiliary point. So we're actually intersecting. This is going to be our subgrade daylighting.

Now obviously, if we're in shallow fill, we can't obviously go and create a point out there. So we have to treat it slightly differently. So I'm going to drag in another decision here. And it seems to actually automatically connect up somewhere for some reason.

And all we're going to do is add another decision. Are we in shallow fill? Or are we in deep fill? So to do that is AP3 going to be deeper than P2. So they're the two parameters we need to look at.

So in the decision here, we just need to say is AP3.Y going to be less than Point2.Y. And if it's false-- I always like to annotate these-- we're in high fill. And if it's true, we're in shallow.

So if it's true, we're in shallow. And we come out here. So let me just drag that out. Oop, let me just connect it up first. So all we're doing in the shallow fill is taking down our depth, boxing out, and just dropping a line straight from the intersection point [INAUDIBLE].

If it's false, and we're in high fill, say somewhere down here, we're just going to use a few intersection point tools here just to create a new point, create some links, create some shapes. So what we're got now is pretty robust batter slope tool that's completely configurable. So within these parameters, we also have point code, link codes, that the user can actually enter their own.

So now we're going to apply it in Civil 3D. And we're going to use the concept of a linked corridor. And once again, this is one of those eureka moments when I was playing around with it. I just thought this is awesome. So what we've got here is a road alignment. We've got a road here. that's just using one simple subassembly.

It's just one region from start to finish. It's got conditional horizontal targets. So if we find the target, we're going to put in what we call an SO type drain in Australia. This is just a dish drain, a meter wide 200 mils deep, collects a bucket load of water.

I'll show you briefly what this subassembly does. This is using-- just go at control one. If I didn't like that type of kerb, I've got a list here that I've already created some predefined standards from our libraries. Users don't need to change it. They don't need to muck around with it. They just select the one they want from the list, and away they go.

If the target isn't found, effectively we're in fill. I'm going to put in a standard fill situation. That is actually going to be controlled by standard AutoCAD poly lines. So, yeah, that's this roadway here. What I'm going to do first is I'm going to actually fix up the subassembly and just add it to the list.

I've already got a bunch of them here. I've got my five to one. I've got my two to one. This is

my transition five to one to two to one. What I'm going to do is I'm going to introduce the concept of a subsidiary string. Has anyone actually used invisible links before? Yep, a couple? Cool.

So what we're going to do is add a generic link, an invisible link. I'm just going to call it the point code EH, because we use two string labeling conventions. We're a bit restrictive on that. I'm just going to add it to this point here. So you can see it's just created an invisible link. That's going to be the hinge point where we're going to start our new corridor off the hinge point of the main line corridor.

And I'm going to add a AU2015, the batter transition subassembly. I'm just going to hook it onto that point there. I can come in here and say I want to two to one to five to one, which is actually what I want to do. If I want to make it 10 to 1, I could do it that way. So we can plug in whatever values we want.

So that was two to five. So now when it comes to modeling-- just pull this across. How good is it, we can actually move view ports now? Man, I screamed about that for years. What we're going to do now is we're going to create a link feature line on this hinge point.

It runs the whole length. And we're going to link it, make it dynamic to the corridor. So any changes we make to the corridor are going to be rippled through to the batter slopes. So from feature line, credit line from a corridor-- I'm going to give this a name, MC10 LHS Hinge. Always good practice to do that, but I always keep forgetting. I'm going to use the current layer. And I'm going to turn off smoothing. I don't want it automatically smooth. I want it to actually match the facets of the alignment.

Make sure we create a dynamic link. That's critical. So there's our auto corridor feature line. So what we're going to do now is add to the-- I've already generated [INAUDIBLE] the corridor off here. I'm going to come and select it. I'm going to add a region. So I'm just going to start it from here. I'll run it through to about-- that looks good.

I'm just going to do a simple left hand side two to one batter slope. OK, existing ground, that's our final daylight surface. These two here are substrings. So all we're going to do here is use the feature lines that we've just set up, the left hand side hinge. I'll do that for both horizontal and vertical. And I'll Now select it from the drawing.

OK. So what we've got now is a two to one batter slope running the entire way through, linked

onto that corridor hinge line. The great thing about this is, it's fast. So I can leave my roadway as is. And all I've got to do is must move my grips and start to drag them around.

Now what I want to do, I can see there's my cut fill line. I want to make it five to one through there. I want to just sort of roll it back and then roll it back up into steep [INAUDIBLE], once we reach a certain depth.

So to do that, standard Civil 3D tools. I'm going to split the region, say between there and here. I'm going to come into my region properties and just swap that out from LHS2 to LHS5. Done.

So you can see there, that looks pretty wonky. There's no way any contractor is going to want to see a model like that. Contractors these days, they want everything down to the nearest cubic meter. So that's why modeling is actually becoming so much more critical, especially in bin workflows.

So what I'm going to do is split this region again, split it between say there, yep. And same approach, region properties. This one I'm going to go from two to one to five to one. As soon as I go OK, you can see that there's the transition between two to one and five to one.

Now you can see there, I'm outside the boundary-- let me just turn this on-- big fat red line, outside the boundary, no good. How do we control that? We pick up the grips and we move it.

So we've got complete control over that. You can see how fast that's updating. So we don't have to regenerate the corridor every single time. We're only taking care of that one component. That's the part that you're going to be doing most of your work in.

You can see here as well, there's my SO drain. It's actually finishing in a cut. Obviously that won't work. It's not going to drain out. We need to tail it out. So in this case, I'm going to take this and just extend it beyond my cut fill line. Now I'm going to regenerate my MC10 main line.

You can see now, I've actually pushed that out. It's obviously out of date now, because that linked feature line affects what's going on here. Just rebuild that. And we're done.

I can do the same on this side. I'm just going to transition it once again between five to one and two to one. So once again, split region. The good thing is, I'll show you another good way to do this. Region properties, and this is going from five to two.

So once again, grip drag. You could come into your corridor properties and enter property changes. But, honestly, this is just so I quick, it amazes me. What I can do now, I'm back to two to one here. If I want to go two to one to five to one again, I could drag it across and keep splitting. But to be honest, a quicker way I find is just grab the corridor, copy a region. There's my two to one to five to one. Run it between there and there, done.

If I want to do it again, there's my five to one. Copy, paste, done. So once it's all set up and in place, you're good to go. You're firing.

Just to show how it looks from a perspective view, I'm going to risk it here and go orbit. It didn't crash, yeah. Yes, people have done that. Haven't they? So we'll just sort of pan down here. You can sort of see how it looks. So there's our five to one sort of transitioning back in.

You can see it's really smooth. It looks great. We've even got our daylighting of subgrade through there as well. So volumes, no problems.

Anyone seen that before? Does it make sense? Yeah. I liked it. I thought it was really good. I've actually shown a lot of people in Australia this technique, and they really like it. It's saving them a lot of time.

So just going back to the slide show, there's the formula there. It's just a little basic description on how it all works. Don't really need to know how it works. If you've got the formula there, you can just repeat it over and over again. And you'll be right.

Just a nice shaded view. Looks pretty.

OK, we're halfway. So what we're going to do now is take the concept that we've just looked at with the transitioning, and we're just going to take it up an extra step now. So we've got a transition from point A to point B across the region. What we're going to do now is take a bit more control over that.

Do people here model pram ramps and kerb laybacks using feature lines? Sorry?

**AUDIENCE:** [INAUDIBLE].

**ANDREW** Pram ramps? Sorry, am I speaking too fast?

**MILFORD:**

**AUDIENCE:** No.

**ANDREW  
MILFORD:** Must be my accent. So yeah, kerbs and pram ramps, all we're going to do is take the concept of that transitioning we looked at before and we're just going to apply it, almost mirror it, over each end of the corridor region.

And so we'll jump in and have a quick look at this. We're going to use the exact same modeling techniques as we just saw then with a linked corridor. So we can actually just move along and place kerb laybacks or pram ramps wherever we want.

This is a typical pram ramp detail that we'll be looking at. Once again, RMS-- I don't think I explained that before. That's the Roads and Maritime Services. That's a state government authority for New South Wales in Australia. So these are the guys that call the shots. So we've got to do what they say.

So we're going to look at this top left corner here. So effectively our region is going to start about here and finish about here. And we're going to transition up at a nominated offset, or a nominated distance, which we're going to let the user configure as well.

And then we're going to provide a full width for x amount of distance. And then transition it back in at the other end. And it's going to be all dynamic. It's going to be all built as part of the corridor.

This is just a quick example on how it looks. And you can see, it's not that complex. I've seen some of the standard subassemblies that ship with Civil 3D. And they just blow my mind. I mean, I like to keep things simple. But it does an awful lot. It's pretty handy.

So we'll just jump into-- back to the subassembly, pram ramp. OK, once again, we got versioning. This one's got a few more parameters going for it. We've got, effectively, three links.

So we've got our gutter line. I've got to get used to the terminology. We call it the face of kerb. Then we're going to have a top and a back string. So this string you're seeing here at the moment, currently in, it's our face of kerb. It's basically the dish that collects the water.

This is the only gotcha with this sort of pram ramp assembly-- and I'll explain it in a bit more detail when we come to model-- is I need to actually have a target offset to hit that first line at

the origin here at P1. And I'll explain why when we get into it. It could be just a shortcoming of the Subassembly Composer. But it still works.

So there's Point 1 and Point 2. That's just using these lip values that we've specified up here, lip x, lip y. And all it's doing is just, it's just a point. It's the lip line x value and y value. So we're actually catering for both horizontal and vertical transitioning at the same time.

All we're doing here is just setting up some variables. And we're just setting up the deltas, just like what we did with Slope 1 and Slope 2 before. We're just specifying the deltas between X1 and X2, Y1, Y2.

I'm not going to type this one out. But I'll explain what it's doing. Before when we say we're doing the baseline station, we worked out a percentage distance across the entire region. What we're doing here is saying, if our current baseline station is less than the region start plus that transition value-- and I probably should explain that one. Just over here we have this transition value. That's the distance from the start and the end. We're going to be transitioning across.

So we giving the user the control. They might want 1.3, they might want two meters, they might want five. So it's just making it that little bit more flexible.

OK, so if they're in that first portion, or if the current station is from the baseline dot region end minus that transition value, so if we're in the front 1.3 meters or the back, so if we're true, we're going to go and launch off to the left here. And we're going to-- just bring this across-- define our transition variables.

If it's false, we're actually in the middle. We're not doing any transitioning any more. We're going to use our final X2 Y2 values. So we're just going to branch off and just do the full ramp over here. And that's just simple, there's just three simple links in that one. Sorry, two points, one link.

So we bring that across to there. So we're in the transitioning variables. And we're going to branch off to the left and follow through there.

So what we need to do next is add another condition. So we've actually determined now, OK, we're in the transition off to the left here. Now we need to work out, are we in the leading part? Or are we in the trailing? So to do that we're just going to add another decision in here. --that automatic hook up. So I'm just going to pop this open.

And this one is, if the current baseline dot station is less than or equal to baseline dot regions start plus this value over here, called transition. So effectively, if that's true, then we're actually in the leading, we're in the first part of that transition. So we're going to say leading. And if it's false, trailing.

Once again, always nice to annotate your subassemblies. If someone wants to get in there and edit them, they can at least follow the trail of breadcrumbs to get there. So we've defined that. We just need to connect that into there. So if it's true, we go to the leading transition. If it's false, we go to the trailing.

So once again, it's the same type of formula that we used in the previous example. We're just taking it that little extra step. So I'm just going to lead it out into the trailing. So for the leading transition, which we'll focus on, all we're looking at is the variable top X variable top Y. So if we go back to the slide show, all we're looking at is retrieving this value here and this value here. But they're the two that are actually transitioning across that area through that 1.3 meters.

And so the formula is exactly the same. So all we're doing is saying, if the baseline dot station minus the start station divided by that transition value times the delta and adding it to the initial value, which is over here, top X1.

I've said you've got all of this. I don't know if I've annotated them quite as well as I could have. But it's all in the documentation anyway.

So from this lead transition, all we're going to now is-- oh, just to show. In the trailing, it's exactly the same, exact same variables we're editing. So we can-- all we're doing, saying leading, trailing. Instead of saying region start, we're just taking this from the region end and just working in the back portion not the front. Same formula, just flipped.

And then what we're doing is coming down here and connecting on. So there's our first link, P3 and L2. So that's going to go from this out to this. And those values are all specified here, top X1, top X2. So it's going from 0.03 width to 0. And it's going from 0.15 to 0.

In honesty, we probably don't need to put those parameters in. But we've just kept them in there for flexibility. There's our back string. So you can see I've actually only added a point there, not a point with a link. The only reason for that is we're calculating this point from here, not from here. So then I just put another link in there to connect that up.

And then all we do is just finish it all out, build it solid. And that there is just a series of links that sit underneath. You can see, the great thing about Subassembly Composer, you can either select a link and it'll take you there. Or as soon as you select it, it highlights onscreen. So you've got some good visual feedback. You know what's going on in our final shape.

Now the last thing to do here is, we've handled all of the transitions. What we need to do from here is back at the start, where we said, OK, if we're not in a transition, we're just going full width. I just need to run this one, bypass all of that decision making process, and just connect it in there. So everything's all connected now. It's all complete.

This stuff doesn't take long to pick up, by the way. I only started using this about, maybe, two years ago. There's our exit ramp. That's going to go to this drawing here.

OK, so back in Civil 3D, we're looking at exactly the same concept we were looking at before. We have a lane here that's using a really simple single land subassembly, six meters wide. The only difference here is we've got a pedestrian refuge island here. And we want to put some pram ramps in across here. I've got an offset alignment at the back because I know my verge is going to kick out behind it.

I've also got these red lines here, which are going to illustrate some property access lines. And we're going to start looking at putting in some kerb laybacks in there as well.

In this case I've already gone ahead, and I've already set up an automatic corridor feature line on this point down here. It's using the linked corridors. Once again, it just gives us that flexibility if we decide to move our property access kerb layback. We can just slide it along. We don't have to regenerate main line corridor as well as the others.

So I just need to come into my subassemblies here. There's my left hand side kerb. We take a quick look at this. We've got our invisible link, which I've called the substring. We've got our kerb line, the verge string, and the batter transition. Do you guys call them batters? Daylight?

**AUDIENCE:** Daylight.

**ANDREW** Daylight, yeah, we call them batters. I should have probably said that from the outset.

**MILFORD:** Someone told me that. They'd never heard of it. And I thought, hmm, OK. I just get used to it.

So what we've got. I'm just going to copy all of this. So there's our left hand side pram ramp. I'm going to take the pram ramp subassembly and just do a replace. So you see in the

parameters over here everything that came in from Subassembly Composer.

I've said, a lot of values in there we can change. For our standards we don't need to. You can. You can always do it and just experiment and play around with it. But to apply it, all I'm going to do is take the left hand side kerb. I'm going to run it through the entire length. So to do that I am going to select my corridor line here, which is my left hand side kerbs, select outer region.

I'll do it from near enough the start. And I'll take it right up to the intersection line there. And it's kerb, left hand side, that one. Batters, the substring, is going to be the feature line, which I know is already on a layer-- much easier to do it by layer than by selecting on screen-- left hand side lane edge. Once again, width and elevation.

OK, so there's our kerb and gutter line you can see in there. There's our-- gotta get you terminology-- flow line. Yeah, we call it the face. Horses for courses.

So what I want to do now is come in here and do exactly what we did before. We're going to break it, and we're going to just apply the new subassembly in there. So I'm going to use the pram ramp in here. I'm going to select this. I'm going to split it, either side.

Now you can see here. I've got this-- it's probably a little hard to see. I'll show it anyway. This is what I was getting back to when I said I needed to assign a width target to get this working. Because my frequencies in Civil 3D are not set to 1.3 meters. They're set to five or one or two.

I need to actually add additional points manually to get that sharp edge of the pram ramp. It's a bit of a nuisance, but it works. So there's my polyline. I'm just going to swap that piece out, using region properties, RP. And it's kerb return fillets with the pram ramp. No it's not. It's kerb pram ramp.

Doesn't look like much at the moment, because I haven't actually set the targets up correctly yet. First thing, substring, I need to come and reselect the laneage I'm going to specify the verge string at the back here. I do have an offset alignment sitting there. Because I know that I just need to widen out to cater for the minimum for path width. And I know it's that string there.

It's this one here that we need to actually target this pram ramp subassembly. I need to specify that feature line. So I'm going to select that line from the drawing, that one there. And it's just a 2D polyline. Once I go OK, so there's our pram ramp subassembly sitting in there.

I'll just flesh out either side of it just to make it look a little nicer. I'm just going to edit the

targets here. Select that. I'm just going to hit the verge. And I'll do the other side as well. Probably should have done that first.

OK, so that's a pram ramp sitting in there. So if we flip into a side view, you can see it's actually doing exactly what we want it to.

So we get that nice clean geometry. It looks really smooth. And we can easily just slide the polyline up and down. We can then just move our regions around.

**AUDIENCE:** Would that work just as well on the kerb?

**ANDREW  
MILFORD:** Just about to show you. Here's one I have on the kerb. Now I'll show you how easy it is to actually move them around as well. Same thing applies here. We're going to split the region there. I'm going to split it at that point. You see I've got a few temporary lines in there.

But that's OK. They're on non-plotting layers. Region properties, going to change this to kerb return, to pram ramp. OK, once again, that looks a little iffy. I'll set my targets. And I need to set that target to that polyline. And OK, I think everything else is good.

Actually, I'll do the verge while I'm here. Select from the drawing. That's just a 2D polyline. It just slopes back at 2%. OK, OK, you can still see here it hasn't worked. The trick for that is, you need to actually select it, come into your Edit Frequencies. And I need to make sure that my offset targets are using offset target geometry points. Otherwise, it just won't pick it up.

I could do either side there, which Yeah, I probably should. Just edit the targets there. Just make sure I hit the verge. Select from the drawing. OK, I'll do it for the next one just because I like being complete. Nope, that one there.

And OK, OK. So just looking at it from a perspective view, yeah, once again, the geometry in there, it's nice and clean, wraps around nicely. So I've actually moved them around. All I'm doing in this case is I'll take this polyline. I'm just going to rotate it around the center of my kerb return there. I'll say rotate, say minus five degrees.

All I do is pick this up. I'm going to select that and snap it to there. And take that point, snap that out to there. So that's how easy it is to manipulate and move it after the fact as well.

Same applies for a kerb layback. Kerb layback itself-- just coming back to the slide show-- it's pretty much the same as the pram ramp. Only it's got an additional layer of decision making.

Because if you look closely, this is a typical kerb layback detail.

We've got a transition that runs from here, to here, and also to here. So this top string runs in, I think, about 700 mil in Australia. But then there's another 400 mil on top of that for this additional step. So in the subassembly-- I won't go into detail on this one-- you can see, we've got the exact same configuration as we had in the pram ramp. But we're just adding some additional controls for that additional offset. And all it's doing down here is actually creating an additional string to actually connect those two up.

I'll just jump back into Civil, and I'll just throw one of those laybacks in just to show what they look like. I've already got the set out there. I may as well do it, split, end, and region properties. And this is where I'm going to use the kerb layback.

Once again, edit targets just to make sure I'm picking up that right string there. And I think it's all on a layer. Mmm, that looks good. There we go. So the only real difference there is we're transitioning this top string here, this red one, between that point there and there, which is about 700 mil in. But the back string is going from there up to there. And then we're actually introducing this new string in here.

So that's the only real difference between the two. Concept is the same though. So I always like to just spin around it, just to make sure it looks right. If it looks right, it usually is.

So in this case here, I've actually taken out the foot path strings, sidewalk strings. And we can just design in some property access from there.

That's just a quick shaded shot of the finished product there. So, yeah, great thing about that, unlike using feature lines, they're dynamic. They're part of your corridor. You just got to drag your grips around.

You can pretty much move them. Stretch them. Make them as long or as short as you want. And most importantly, they just stay linked to your alignment. So that the rework time is just so much less than using feature lines.

OK, home straight, Exercise 4, Multiple Batters and Benches. This is where we're going to combine everything we've done so far. And we're going to look at the concept. We're going to bring it all together.

We're going to use auxiliary corridors. We're going to use linked alignments. We going to use

custom subassemblies. And what was the other one we're going to use? Oh, we're just going to use some offset alignments.

And we're going to build it all up. And we're going to do a really complex batter slope configuration. Does anyone do major highways design here? Major roads and highways? Cool.

This here is a photograph. I worked on this project about 18 months ago in industry, when I was working for Jacobs. This is on the mid north coast of New South Wales. From it's called Nambucca Heads to Urunga. Really nice part of the world, if you're ever up that way. I wouldn't drive out there at the moment. The temporary works are horrible.

You can see this cut here is approximately 35 meters deep-- huge. Don't know how much dirt we pulled out of it. I can't remember the number off the top of my head. 35 meters deep, varying cut slopes. Due to geotech conditions and a lack of fill throughout the rest of the project, there was a lot of complex earthworks modeling going on in this area. This took the designer of this part-- I didn't do this part-- but it took the designer about two weeks to model up the earthworks in this, using some other software, which shall remain nameless.

But due to the geotech conditions, we couldn't have constant batter slopes the whole way. It made the modeling process really difficult. So we started out with each end being at three to one. We went from three to one to 1 1/2, to two to one, even up to 0.75 to one. And they were varying all over the place.

So we had this multi-bench configuration all doing this crazy, crazy stuff. And as you see there, it's huge. So what we're going to do is, my aim was to take it from the old modeling package, bring it into Civil 3D. And I wanted to make one subassembly that could do it all.

And this is it. This looks pretty complex. But in actual fact, it's the same thing, just repeated each time for each bench. So we'll jump back into Subassembly Composer, and I'll show you how we built this one out. This we actually do need to utilize the concept of offset end elevation targets. And there's quite a few of them in there. But the end result, it's just incredible what you can do with it.

Oh, I've just given that away.

So you can see here, all of our targets are in place. You can see there's quite a few of them

over here. We've got multiple surfaces to hit. We've got quite a few width, five width targets and five elevation targets.

Now I could have actually gone and made more. I could have made six layers of benching. I could've made seven. I could've made 10. But to be honest, once you get past five you're going to be tunneling anyway. So five was good enough.

So the first thing we need to look at with this is the most simple decision, sitting at the top of the tree, are we in cut or are we in fill? And that's all this is doing here. You can just go back to my input-output parameters.

OK, so at our base point, our hinge point, we had AP1, which is an auxiliary point. I haven't actually explained that. If you haven't used Subassembly Composer, an auxiliary point is a point you use just for setting out other points. They don't create geometry. They're just there to help you drive further decisions and get more control over it.

So this is an API function here. So AP1, distance to surface, Surface 1. Now Surface 1, in this instance, is our final daylight surface, not our intermediate benches. So if the AP1 is greater than 0, if it's true, we're actually in fill. In which case we just branch out over here. So I can take my surface one, just do a bit of a test. So we're obviously in a fill condition there.

And that fill slope is four to one. I can maybe make that two to one. So you can see immediately what it's looking like.

If it's false, we're in cut. So if we're in cut, we're going to go and create a point that just sits on the hinge. That's our starting point. So if we're in cut, you can see here. I'm just going to take surface one back up again. And that's all we have.

The next thing we need to do is the first of two decisions. It's only two decisions to generate this entire thing. We take this, and we're saying here, Surface 5, this is our bottom bench. The one at the hinge point. If the surface is actually valid, i.e. It exists, and if the distance between Point 1 and this Surface 5 is less than or equal to zero, if it's true, then we're going to come out and do another test. If it's false, that means we have something like that.

So that surface is actually, it's no longer valid. We don't need to use it. We're actually too close to the final surface in order to get it to work.

So if it's true, we come out and we're going to create this auxiliary point, AP2, which is sitting

over here. Now all AP2 is, is we're using the cut slope down here, which is going to be three to one, just as a constant three to one slope. And the delta X is purely a value driven from that slope multiplied by the batter height.

So that batter height is the height we need to go before we hit a bench, effectively. So once we have that in, we have AP1, AP2, Surface 5. I need to add another decision here. And I need to test if AP2 is achievable to hit the final surface.

So I just need to add another decision in about here. Get rid of that. So AP2-- I just need to enter the expression in here-- so if AP2 dot distance to surface-- and what's the surface we're going to test to? We're going to test it to Surface 1. And if it is actually greater than zero-- so if this point here is actually-- let me just go zoom out a little bit here.

So I'm going bring Surface 1 down a little bit just so I can show how this works. So we're saying, if this point here is sitting above the final surface, we're just going to batter to the final surface. So if it's true, we're going to branch out here and say batter to the final at that Surface 5 cut slope.

If it's false, and it's actually a little bit too far out, we're not quite there yet, we're going to cut branch out to the right. And we're going to actually introduce our first bench. In which [? place ?] we come out here, connect, and we create this bench scenario.

So within this bench, we've got 0.2 sitting up here with the first link. It's using the first cut slope. But this is where we have the option now to override it with our width and elevation targets.

All we do from there is copy paste, copy paste, copy paste another three times. And once that's all hooked up again, we end up with a pretty complex looking batter slope configuration. Probably run Surface 1 back up. So we can actually control every single element of this. both height and width, if we choose to override it.

So I'm just going to jump back into Civil. I'm going to close some of these down and open up some new ones. I'll just open them all up while I'm here.

So, yeah, the great thing about-- the only problem with the previous approach, when I mention that joint batter slope cutting before was the guy who spent roughly two weeks modeling it up. Two days after he actually had it modeled, he was told to change it.

It wasn't a lot of fun. And it was all done via code. And I think there were about maybe 1,500 lines of code to actually generate the batter slope. Whereas we're doing it here with one subassembly.

That's interesting. OK, I'm just going to pick up some-- I'll close that one off.

OK, so the first part of this-- let me just get to the right drawing here, in Drawing 1.

All right, you can see here, there's a quick cross-section we have. It's actually, you can see there, it's pretty deep. The main roadway has already been designed. That's just using just standard Civil 3D corridors. We've got some median, 10 to 1 median slopes sitting in there. Actually, I think they are six to ones.

The level of this southbound control was actually driven from a temporary planes corridor that we spoke about earlier, the auxiliary corridors. What we have in this instance here, we have our linked corridor feature line on the side, which we're going to run our earthwork string from.

So the first thing I'm going look at is creating an auxiliary corridor. I know I'm going to be using seven meter benches. But I'll need to be able to control it using the width strings.

So in this case, I have some assemblies down here. I'm going to use this thing called seven meter benches. And I'm going to create an auxiliary corridor that hangs off that hinge point. So to do that, I've already got a substring set up, as I like to call them. I'm going to create a generic link, width, and slope.

Now this is a pretty deep cutting. So I'm going to need to go out a fair way. So I'm going to make it about 150 meters wide. Point code I'm going to keep the same. But I'm just going to call this link code plus 07M.

Because what we're going to do is we're going to create a whole lot of surfaces from the one corridor, at seven meter intervals. Now, I probably should have made that 0%. Otherwise that's really not going to work.

And then I'm going to use AutoCAD move command. I'm going to move it up seven meters. I can select this again, copy, selected, change the property. This is going to be my 14 meters. I'm going to copy it again, make that plus 21. And once more, I should have chosen something with only two benches. But it didn't look as good. Gotta go for effect.

Plus 28 meters, get rid of the comma. And we're good.

OK, and lastly, I just need to bring in that batter cut benches. I've already got it on the right hand side. If I just do daylight multiple surface, you can see I've got the hinge point there. Attach it on.

One subassembly, that's it. We've got all the control we need sitting in our targets. In this case here, you can see the parameters involved. Once again, there's a few, but it's not unmanageable. Let me just pop that open.

My final fill slope, I'll make that three to one.

So you can see here, we've got our Surface 1 cut slopes. I've chosen three to one, 10 to one, negative 10 to one on the bench slopes, and made them 4 1/2 meters wide.

If you're doing a simple bench slope, you don't even need to worry about targets. I'll show you what it looks like in just a second.

So back here, I need to actually go and edit the corridor. I've got a working seven meter bench is here. I'm just going to select it. And I'm going to add a new region between say here and here.

I'm going to use that benches model. So this is the auxiliary corridor. This is what we're going to use to drive the elevations of the benches.

Make sure that substring is hooked up. Left hand side hinge, and OK. I just need to do one more thing. I'm just going to turn that on. OK.

So all it's doing is creating this big temporary working surface. So if I pan around, you can see there are seven meter benches, one, two, three, four.

All I'm going to do now is start to create some temporary surfaces through that. I don't actually need the frequency that lodge either, to be honest. I'll just leave it.

So in corridor properties, we come to our surfaces. Those link codes that we mentioned before, we're going to use those to actually drive the individual unique surfaces. So I'm going to do a seven meter, make sure I give it a nice logical name, plus 07M.

At the seven meter benches, we're going to do it again for the 14. Bang, bang, bang, let me

just pop them all in. 14 add, 21 add, and finally 28. Apply it. Plus 14, plus 21, and you get the idea.

So all that's doing now is creating four working surfaces that are now linked to our parent allotment. And now we're ready to go to the next part, which is 0 2 if I can--

For some reason, my layer dialog box seems to be have glued itself to my screen. --two.

This instance here, what we're going to do is drive the geometry of the benches through the use of the auxiliary corridor, which we've already set up. And we've generated the four surfaces-- 7, 14, 21, 28. In this case, we're going to use an offset alignment. What I've done in this case, I've already set one up on the hinge.

Because what I want to do is, I know over here, just doing some simple CAD work, that I've got some batter slopes already sitting in here. My 0.75 to one, I know is 5.25 meters from there to there. It's just math. Three to one is 21 meters, and my two to one is 14.

So this way you just need to do a little bit of maybe manual offsetting, just offsetting alignment strings. So in this case, I've got an offset alignment sitting 12 1/2 meters from my control line here. The reason I'm using this offset alignment on the hinge is because when I create these bench offsets to control the horizontal targets, should that width change, they will actually move with it. And I don't have to keep readjusting. That's the only reason I'm doing it this way.

So in this case, I've got my offset alignment running through here. And I'm going to add some offset alignments. I'm just going to add four of them to the left at 21 meters, approximately, offset.

We're; going to be editing these a bit later. Nothing to the right. I'm just going to accept the default names. So we have four offset alignments that are each going to represent one of the benches that we're going to be using.

So all I need to do now is add that surface profile to each of those respective benches. So the first bench here, I'm going to add a surface profile of working bench seven meters.

The second. This is my 14 meter, 21, and last one, 28.

So we've taken care of now the horizontal. We've just taken care of the vertical now. So we can now actually start to look at some proper targeting.

So we're back in to the final doing now. I've got a few construction lines here. Like I said, only for the sake of the exercise.

We're now going to add the subassembly, or the assembly, to the corridor. That's the corridor that's now built off the MC10 main line. I'll just rebuild that. And I'm just going to select this. I'm going to add a region. So I'm going to add the Select, Add a Region, between say there and there. And I'm going to use the batter cut benches left hand side.

OK, existing surface for all of these targets, we don't need to worry about those. I've got the substring set up. No I don't. I thought I did. --Select by Layer, OK. Let me just go back in there into the properties. Just turn that on. OK, so you can see straight away, we've got the benches in there.

That's just using the default values straight out. So if we go into a perspective, you see it looks pretty good. It's doing. We've got three to one cut slopes at all the different locations. If I were to come into my subassembly and maybe change the front slope there, Bench 5 cut slope.

If you want to make that say, 0.75 to one right from the outset, just rebuild. You can see the rebuild times are a lot quicker as well, because of the linked corridor method. You can see, that's actually really quick. I'll put that back to where it was. So I'll just undo that. Let me just change that back to three to one, rebuild, cool.

But what we want to do now is, that's my three to one line. You can see it's three to one up roughly that point there. So that's obviously 21 meters away. Because it's introduced the first bench through the subassembly.

What I'm going to do now is actually start to split-- get rid of that-- is I'm going to split this offset alignment that we've used and then just target it. So I know that these points here-- this point here is offset roughly 10 1/2 meters from the hinge. That's going to give me a 1 1/2 to one line. This purple one down here is my 0.75 to one. And you can see I'm going to have to marry it back in up here somewhere.

So I'm just going to do my first split, and bring it back to roughly that point there. I'm going to split it again and marry it up to say there.

For a first pass, you don't need to be exact. You just need to start getting the geometry in place. Then we can fine tune it later. And that's our 0.75 to one. So what I'm going to do here is just bring that down. I can just select that and make that minus 10.5. There we go. So I've

got a transition here. I just need to bring that in a bit. You can see it's linear.

I know in Australia they actually like to flair them just to create a much more aesthetically pleasing look. The urban designers love that stuff. Contractors couldn't care less, I'm sure.

Curve reverse curve-- so in this case, to get that flair, all I need to do is change my curve one radius to one. And we just get that nice smooth, flared look in there.

In here, I'm going to do the same thing again. I'm going to split it. In this case I'm going to do some reverse curves. So linear, curve reverse curve, I'll make that exactly 50 meters, because that's what they usually ask for, and maybe 125.

There's our point 0.75 finish. I'm just going to take this out to there. And I'm going to do a curve reverse curve again, once I select it and go to Group 2. In this case, it's not Curve 1. It's going to be Curve 2 that is one meter. Obviously that's sitting out a little bit too far. Because I already know that it's going to finish up somewhere around that area.

I'm just going to bring that down there. And I'm going to slide it along to roughly that point there. OK, how are we doing for time? 10 minutes, excellent. This used to always run over when I was practicing.

So all we need to do now is start to edit some targets. So the first target, width elevation target, I'm going to select that alignment string, offset seven meter bench. And do the same for the elevation target-- Profiles, MC10, that one there, OK. So there's our first of the benches.

The great thing is, this is now completely dynamic. Once that's been set up, we can just simply push and pull the vertices, and we can adjust this to our heart's content.

Hello? Look there, I've gone the whole way. 10 minutes to go, it decides to freeze on me. That's cruel. Come on. How are we looking there? Right. Let's just fire that up again. Shall we?

So all we have to do from there is just repeat that process and just assign targets for the rest of the benches. I think we'll have enough time, I'll-- no, actually no, probably not. I'll just go to the final drawing.

OK, Open,

So this is the final drawing. I've actually uploaded all of these drawing examples into the data sets. I'll probably put the slide show up there as well sometime in the next day or so.

I'll just do a quick import. That's a bit better. No, that's not.

OK, so you can see here. This is actually the final version of the drawing with all of the benches and offset alignments assigned. You can see there. It looks, it just looks awesome. I mean, it just looks smooth. We're going from three to one, to 1 1/2 to one. If I turn on the final working surface, you can see there. There's our 0.75 to one coming back in.

I said the thing I love most about this is, if we decide to move our alignment up or down, left or right, the offset alignments will stay linked. We stay linked to the hinge string due to the feature line that's been set aside. And if I ever decided to change this and say I don't really want that to be 0.75 to one. I want to push that back to 1 1/2 to one, I just simply do this. And the entire corridor rebuilds itself.

I can now bring that back up to there. So it's just fast. And it's all in one subassembly. So that's the part I really like about that.

And that's just a nice shaded view of the entire thing. You can see some nice gradual sloping in there.

So that actually brings me to the end of the formal part of the presentation. If you like what you've seen here today, by all means please vote, either via the survey station or via mobile or email. You actually have a chance to win a pass to 2016 AU next year.

So just a reminder, every single one of these classes, they are being recorded. The material is all up on the website. Yeah, so with that, I'd like to thank you all for the opportunity to present here today. It's been a lot of fun. So with that, if there are any questions, I'm more than happy to answer them while I'm here.

[APPLAUSE]