

## **“Drive-Through” Parking and Other Ways to Add Realism to Your Airport**

### A Tutorial for Microsoft Flight Simulator 2004

Microsoft Flight Simulator 2004, or FS9 as it is commonly known, does several things by default that I wanted to eliminate when I developed CYYJ (2006). They are:

- AI departing from a parking spot are pushed-back,
- FS9 draws a “fillet” at each corner where a taxiway intersects a runway or another taxiway, and
- when taxiways with centerlines intersect - even if only one of them has a centerline - FS9 draws additional centerline lines towards all possible destinations.

I wanted my CYYJ to have none of these characteristics. Aircraft at the real CYYJ use “drive-through” parking; they do not push-back. Taxiway “corners” at CYYJ are typically sharp, with triangular asphalt “gussets” being laid down where a wider turn is to be permitted. And finally, the “spider-webs” of taxiway centerlines at FS9’s intersections is not representative of many intersections at CYYJ.

### **BACKGROUND**

The Help feature of Lee Swordy’s AFCAD 2.21, the standard tool for airport development, states “because of limitations with the AI, it [drive-through parking] rarely results in the kind of behaviour one would hope”. However, from correspondence with Jon Patch, I discovered he and Holger Sandmann, both well-known freeware- and payware-scenery developers, had successfully implemented “drive-through” parking at their CYWH water airport. A series of posts by Jon and Holger in the AVSIM forum outlining the key factors in their implementation appears at

[http://forums.avsim.net/dcboard.php?az=show\\_mesg&forum=123&topic\\_id=28566&mesg\\_id=28566&listing\\_type=search](http://forums.avsim.net/dcboard.php?az=show_mesg&forum=123&topic_id=28566&mesg_id=28566&listing_type=search).

In summary, those factors are:

- separate paths must be provided for arriving and departing AI;
- the departure path must connect the outbound side of the parking spots only to the outermost nodes on the runway through hold-short nodes within the prescribed distance (225’) of the edge of the destination runway; and
- the arrivals path must connect the inbound sides of the parking spots to all intermediate points on the runway where traffic is to exit.

I later learned that this general approach is commonly referred to as “plumbing” or “pipelining” – due to the use of parallel taxiway networks that, in plan view, suggest the hot- and cold-water plumbing in a house – suggesting that Jon and Holger were not the only ones to use this technique. (For consistency, in this tutorial I’ll use the term “pipeline” to refer to the individual taxiway networks, i.e., arrivals pipeline and departures pipeline.)

I examined Jon's and Holger's CYWH implementation. Since it was a water airport, there was no defined taxiway structure. They could route traffic wherever they wanted. Also, conveniently, all AI parked in one general area. I wanted to apply their principles to a land-based airport with three runways, eleven taxiways and several aircraft parking areas.

Initially, I set up two active invisible taxiway networks (dual-pipelines) – one for departures, the other for arrivals. All AI traffic would use these taxiways, making the visible taxiway network (which I'll call the "base network") completely passive insofar as AI is concerned. While I felt it should be possible to use the taxiways in the base network for arriving traffic, which I later confirmed to be true, I wanted to keep my first attempt as straightforward as possible. To my pleasant surprise, the three-network (arrivals, departure and base taxiways) solution worked pretty much as I had intended. However, "along the way", I encountered a few rather-severe difficulties with undocumented "features" of FS9 (see the following section "Gotchas and Other Fundamentals"). Since I could find nothing in the forums regarding these difficulties, I posted several lengthy accounts of my experiences in the hopes of helping others avoid them. Those posts met a variety of reactions – from gratitude to derision (the problems I encountered didn't really exist according to some). But, a common reaction also was "Why three networks?" when two would do.

Theoretically, two should do. So, I re-implemented my scheme - dispensing with the arrivals pipeline – instead using the base network for arrivals. While eventually, I was successful, I ran into another "gotcha" that had been masked by the dual-pipeline approach.

It would seem that if one is simply implementing "drive-through" parking on an otherwise relatively-standard airport, the single pipeline, (i.e., departures-only) approach is adequate. But, in my case, in addition to implementing "drive-through" parking, I also made major changes to the visible taxiway network, as noted in the introduction to this tutorial. Consequently, my finished airport utilized over 800 nodes, over 500 of which were part of the airport definition. Adding another 100 or so nodes for arrivals to the already crowded base network while avoiding the "gotchas" was much more difficult than creating a third network dedicated to arrivals. As well, given the very congested base network, troubleshooting was, and I felt maintenance and further modification would be, a "nightmare" in the single pipeline arrangement. So, while I was able to make that single-pipeline arrangement work, once I had done so, I quickly reverted back to the dual pipelines. The size of the respective final ".bgl" files were virtually identical.

In summary, if you're just doing "drive-through" parking, use the base taxiway network for arrivals. But, if you plan to make your airport "pretty", do yourself a favor and use a dedicated arrivals pipeline.

In the illustrations below, for clarity, I have omitted the non-essential (to the point being highlighted) nodes on the base network. You may compare the full single- and dual-pipeline implementations using AFCAD 2.21 to view the files "AF2\_CYYJ\_2.bgl" and "AF2\_CYYJ\_3.bgl", respectively, which are included in the folder that contains this tutorial.

## “GOTCHAS” AND OTHER FUNDAMENTALS

As you probably already know, with FS9, things are not always as they seem. Particularly insidious are airport configurations that appear normal in AFCAD 2.21, but don't perform as specified. In several cases, I spent many hours investigating what was wrong and how to avoid such situations (which generally meant figuring out what was going on inside FS9). Unless you want to do the same, be aware of the following.

Node Proximity Difficulties - If two links (taxiway, apron-route or one of each) terminate on nodes very close to one another, all may appear OK on the AFCAD 2.21 display. However FS9 may not create the links as you intended.

When drawing a link, FS9 appears first to check whether there are any other nodes within a critical distance, about 8' or 2.5 m., of the starting node. If there is an older node (i.e., one entered before the specified starting node) within that range, the link simply is not drawn. If there are no older nodes, a link will be drawn – but not necessarily to the intended destination. But, before drawing the link, FS9 also appears to check whether or not there are any other nodes within the same critical distance from the designated destination node. If not, the link will be drawn as intended. But, if there are other nodes within that range, FS9 will draw the link from the specified starting node to the node-in-range which is furthest away from the starting node – irrespective of where you told it to draw. None of this is evident from the AFCAD display. Such failures to draw links results in “broken” taxi routes while drawing to a node different from the one specified potentially interconnects taxi routes. If this happens, it's unlikely the AI at your airport will work as expected.

The most common symptoms of these node-proximity difficulties are AI disappearing immediately after receiving taxi-to-parking clearance or arriving AI mysteriously finding their way to the departures pipeline and vice versa. If you suspect such problems, you may confirm whether or not they exist by enabling the centerlines of the arrivals and departure pipelines and comparing the FS9 display to the AFCAD display. If you have node-proximity problems, the two displays will be different at or near the areas where you're having difficulty. Occasionally, it is a link in base taxiway network link that is not drawn, or drawn incorrectly. That should be obvious from the FS9 display. There's no indication on either the AFCAD or the FS9 display as to which is the starting node and which is the destination. Fortunately, it usually doesn't matter.

Due to the limited sample number, the details above of what I think FS9 does when it encounters nodes in close proximity may be in error. Perhaps it's not the older node or the one furthest away. But, one thing is for certain. That is, when nodes are in very close proximity to one another, FS9 may not do what was intended.

Taxi-to-Parking - Arriving AI stop to seek clearance to taxi-to-parking immediately after crossing the first hold-short node they encounter in the arrivals pipeline after leaving the runway. One would assume that, upon receiving clearance, the aircraft would simply continue on its way. But that's not what appears to happen. As part of the clearance procedure, FS appears to compute the path to parking from the node closest to the aircraft's then-current position – which is one aircraft radius away from a hold-short node. Whether or not the computed starting node forms part of the same pipeline as the hold-short node just crossed doesn't seem to matter. As a

consequence, depending on node configuration, an arriving aircraft could jump to the departure pipeline, another taxiway or even, for example, to a runway edge line (decoration) that leads to nowhere of interest – in which case, the AI will simply disappear.

To ensure that arriving AI stays in the arrivals pipeline, if there are other nodes near your hold-short nodes on the side away from the runway, you would be wise to move those nodes several hundred feet away if possible. As well, the “stopping area” should be bracketed by nodes in the arrivals taxi-path (pipeline or base network) for a distance of about 100’. This will ensure that the closest node to the stopped AI is in the arrivals pipeline. Please also note, if the taxiway makes a sharp turn in the vicinity of the hold-short node, another extra node in the arrivals network may be necessary to ensure the AI tracks straight-away from its stopped position.

Parking Connectors Radius of Turn – While the use of “pipelining” or “plumbing”, will allow you to determine the direction in which an aircraft enters a parking spot, other criteria will determine whether or not the aircraft parks as you intend. Presumably, you want the aircraft to “drive through”. (Otherwise, why are you reading this?). But, the aircraft will still push-back if the radius of turn out of the parking spot is too tight. So, if your AI pushes-back after entering the parking spot in the right direction, chances are you need to enlarge the radius of turn out of the parking spot. A turn radius of about one-third to one-half the radius of the parking spot seems to work in all cases. Similarly, if the radius of turn into a parking spot is too tight, the AI may simply stop without making the final turn.

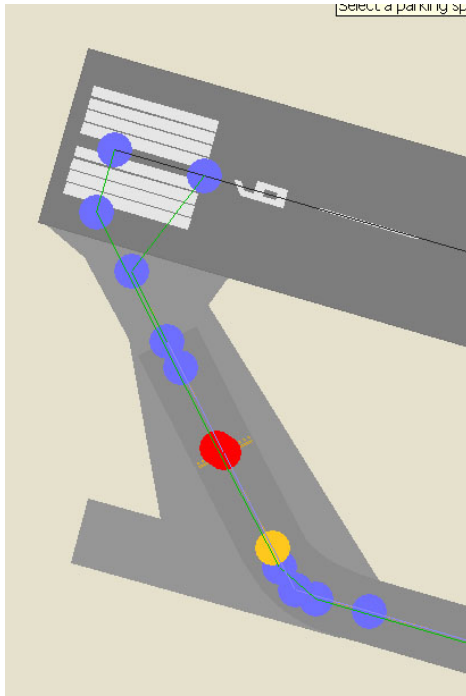
If your AI continue to pushback after you enlarge the radius, there’s one other thing worth checking. If you decompile your AF\_...bgl file (using BGL-XML or some other such tool) and look for the heading “<TaxiwayParking”, you’ll find a field named “pushBack” for each parking spot. This field will contain one of four values, namely: NONE, LEFT, RIGHT or BOTH. The AFCAD 2.21 Help file doesn’t mention this field, nor does AFCAD 2.21 provide any means to control it. I’ve only found a couple of mentions in the forums, where it’s suggested the field is unused. But, to be on the safe side, I set the field to “NONE” in all cases and recompiled the file. Whether it helped or not, I don’t know.

AI Proximity Detection – When drawing pipelines, it would be convenient to use 0-width apron routes. Unfortunately, doing so invites AI collisions and pile-ups, since FS9 does not detect AI for collision-avoidance purposes on 0-width apron-routes. According to Holger Sandmann, apron routes must have a minimum width of 20’ (6 m.) for proper collision avoidance in FS9. Any width larger than 20’ should do. I tend to use link widths of at least 50’.

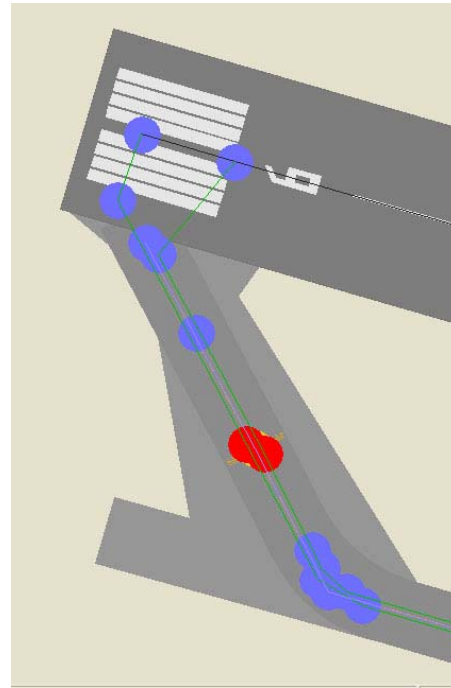
## HOLD-SHORT NODE CONFIGURATION

The single, most-likely source of problems in implementing drive-through parking is the configuration of your hold-short nodes. Shown below are my preferred configurations. Other configurations may work equally well but be aware of the principles involved if you deviate. In the single-pipeline version, the blue line is the base network and arrivals path while the green line is the departures pipeline. Arrival-only links are also green. In the dual-pipeline version, both the departures and arrivals pipelines are green, arrivals being the one closer to the runway.

The arrivals and departures paths each require their own hold-short nodes. In the single-pipeline configuration, since the arrivals path and the base network are one and the same, only two nodes are required at each hold-short point – one of which will be invisible (see below). In the dual pipeline configuration, since both pipelines are offset from the taxiway centerline, a third, centered node is required in the base network for visual purposes - the hold-short nodes in the two pipelines being invisible on the FS9 display.



*Single pipeline, extra arrivals node highlighted*



*Dual pipeline, base H/S highlighted*

The hold-short node in the departures pipeline is placed on the runway side of the visible hold-short node so as to avoid interference with arriving AI (see “Taxi-to-Parking” above). To understand why this still works satisfactorily for departing AI, visualize an aircraft as a moving circle of radius equal to the aircraft radius (which, of course, is aircraft dependent). FS9 stops this circle about 25’ (8 m.) short of the hold-short line. Therefore, so long as the node in the departures pipeline is not more than 25’ beyond the visible hold-short node, the aircraft will still stop on the proper side. A second criteria is that the departures hold-short nodes must be placed within 225’ (68.6 m.) of the edge of runway. You can use AFCAD 2.21’s “H” function to confirm they are within range.

On the other hand, in a dual-pipeline configuration, the hold-short node in the arrivals pipeline is placed on the parking side of the visible node, so as to take the visible node (which will not lead to parking) “out-of-play” for arriving AI. An added advantage of this configuration is that, should an arriving AI require the full length of the runway to stop and, hence, leave the runway from the departures node, it will be forced to the arrivals taxiway network after stopping beyond the departures hold-short node (since the closest node at that point will be on the arrivals

network). The location of the hold-short node in the arrivals pipeline is not otherwise critical since it will be invisible and, in any case, has no operational significance in real airport operations. Its purpose in FS9 is solely to halt arriving AI at a defined point to receive taxi instructions, and the location of hold-short areas is convenient for this purpose. Indeed, you could place hold-short nodes in your arrivals path (for receiving taxi instructions) anywhere on your airport, with AI stopping at the first one encountered.

At each hold-short location, once you have drawn the pipeline or pipelines (procedure described below) and adjusted the location of the visible hold-short nodes, place a hold-short node in the departures pipeline about one-quarter to one-third a node diameter beyond the visible node towards the runway. If you are implementing dual-pipelines, place another hold-short node in the arrivals pipeline an equal distance from the visible node away from the runway. The distance between the centers of these nodes and the center of the visible node must be greater than the “critical distance” (see “Node Proximity Difficulties” above). These additional hold-short nodes will have hold-short lines associated with them. The orientation of those lines doesn’t matter for the moment.

Also, for both single- and dual-pipeline implementations, place a regular node in the arrivals path at a distance away from the runway from the arrivals hold-short node on not less than the radius of the largest AI aircraft to use the airport. For most airports a distance of 125-150’ ( 40-50 m.) should be adequate.

Now, delete the link between each additional hold-short node and the node on, or leading to, the runway to which that hold-short node connects, redraw that link - *from the runway towards the hold-short node* - and set its width to 0. (If drawn properly, the hold-short line will be properly oriented and it will disappear when the width of the link is set to 0.) Typically, these 0-width links will be short and are located in places where it is unlikely that more than one aircraft at a time will be present. However, where the length of such a link becomes extended, as it may in the case of the departures pipeline crossing a runway, one or more additional regular nodes may be inserted between the hold-short node and the runway to limiting the length of the 0-width link. (There is some question whether or not a hold-short node in the departures network serves any purpose in a runway-crossing situation.) It will be necessary to delete and re-draw these 0-width links whenever a node is placed in any link terminating on the respective hold-short node.

## AIRPORT MODIFICATIONS

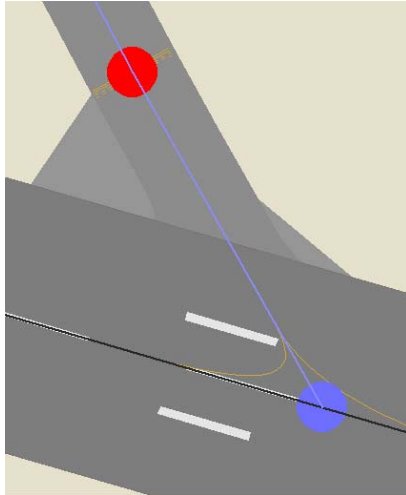
When the goal is “drive-through” parking, it’s tempting to start drawing pipelines right away. However, any pipelines you draw are likely to be temporary if all intended other modifications to the airport have not been implemented first.

Avoiding Taxiway Fillets - Whenever a taxiway intersects a runway or another taxiway, FS9 attempts to smooth the corners of the intersection by drawing fillets. These fillets often are not found at real-world airports and may be eliminated in several ways.

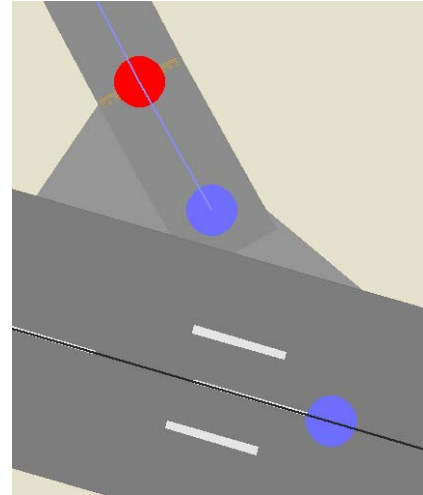
If the intersection is between a taxiway and a runway, the possibilities include:

- dead-ending the taxiway; if there’s no intersection, there’s no fillet – but no AI will find its way through on the base network either (see illustration below);

- decreasing the width of the (black) runway taxi link; if the link is made much narrower than the runway itself, the fillets will still be drawn, but they will be covered by the runway (AI doesn't mind how narrow is this link);
- placing an extra node in the taxiway very close to the runway centerline; again, the fillets will be hidden by the runway; it looks messy on the AFCAD 2.21 display, but it works and doesn't have some of the downsides of the other two methods; and/or
- covering the fillets with an apron polygon.



*Normal taxiway-runway intersection*

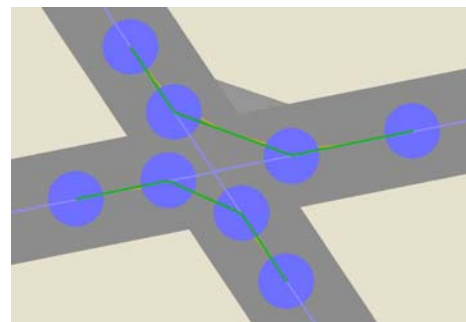


*Dead-ended taxiway*

If two taxiways are involved, the alternatives are similar, except that in the second case, there's no runway taxi link. However, depending on the situation, it may be possible to place a narrow apron route link down the centerline of one taxiway and use it for the intersection.

To help replicate the real-life airport, AFCAD 2.21's apron polygon drawing feature may be used to draw filler 'gussets' and other shapes. These polygons may be bordered with lines (1'-wide apron route or taxiway links with centerlines) and taxiway lights (apron light strips).

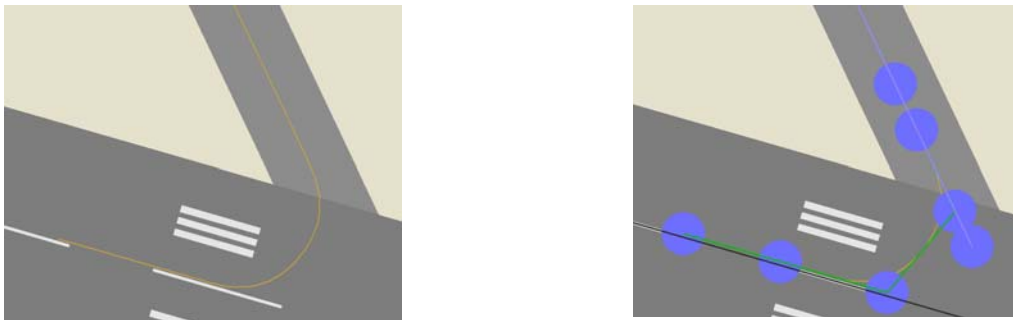
Making Curved Taxiway Centerlines - If you implement a dual-pipeline or otherwise modify your airport, you're almost certain to have to replace portions of runway centerlines or to add new segments.



*Simple curve*

There are two situations to consider. The first, shown above, is where you join or rejoin centerlines; the other is where one or both ends of the curves are “free-standing”. The general technique is the same in both cases; the difference is in the number of segments required. For most situations, a three-segment line should be adequate. FS9 draws such a line as a continuous curve with only the very ends of the line oriented with the starting and ending links. The first and last links should be aligned at the starting and ending orientations; the intermediate nodes should be placed equidistant from the point of intersection of the extended first and last links. The radius of curvature is largely controlled by the length of the center segment.

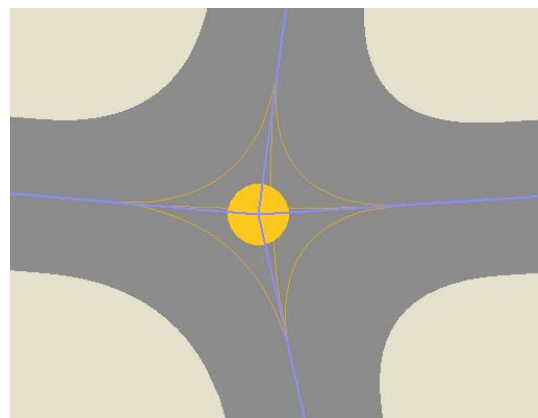
If a line-end is to be free standing, such as the case where a taxiway intersects with a runway, you’ll probably want to add an extra segment at the free-standing end(s) to provide a short straight portion before starting the curve. Generally, the width of the links should match the widths of the links leading into the curves, whether taxiway or apron route links are used – apron route links providing only the line; taxiway links providing both the lines and the surface on which they are drawn.



*Open-ended curves*

Curved taxiway edge markings may be added in a similar fashion, either in place of, or in addition to, centerlines. Straight taxiway centerlines and/or edge markings are easily added using 1’ wide taxiway or apron route links.

Eliminate those “Spider-Webs” - When two taxiways with centerlines intersect, FS9 draws not only the intersecting centerlines but also a curved line from each taxiway towards each possible destination. This may not be representative of the lines at the real-life intersection. There is no way to selectively reduce the number of lines on visible taxiways. The only alternative is to insert nodes in the taxiway to eliminate the centerlines in the vicinity of the intersection and replace the desired missing lines using taxiway or apron route links segments of appropriate widths with centerlines.

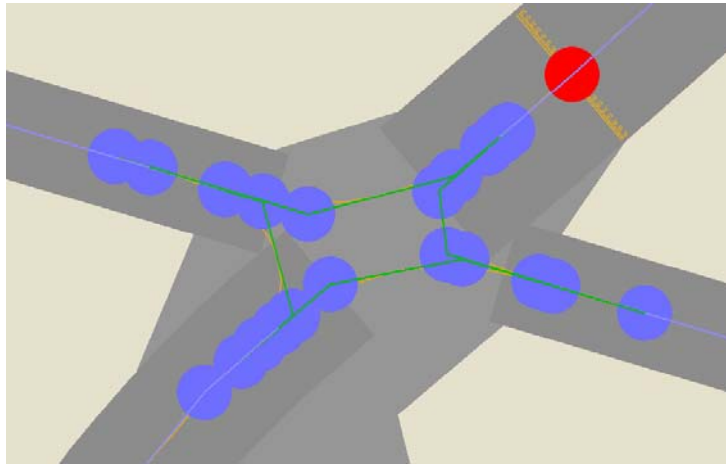


*Spider-web*

Turning off the centerline of a taxiway segment alone may not be adequate. As you may have



already discovered, for a particular link, having selected centerline off doesn't mean no centerline will be drawn. What will happen at the transition point between a link with centerline and/or edge lines on and another with those lines turned off is that the lines from the first link continue for some distance into the second. The only way to avoid this is to also set the width of the no-line segment to 0. But be careful; there's no AI proximity detection on 0-width links,



*Complex taxiway patch involving dead-ended taxiways and an apron polygon (edge lines omitted)*

proximity difficulties, you must ensure there's adequate spacing between these nodes.

Moving the no-centerline transition point further back away from the intersection may help. However, runways, non-involved taxiways and other features will limit how far back you can go. If you can get back far enough, you'll be able to keep things simple. If, however, you can't get back far enough, or the lead-in segment(s) isn't straight or the taxiway changes direction in the intersection, other methods may have to be used. As well, as can be seen to the left, eliminating fillets and replacing centerlines uses a lot of nodes –all on the center of the runway. To avoid potential node-



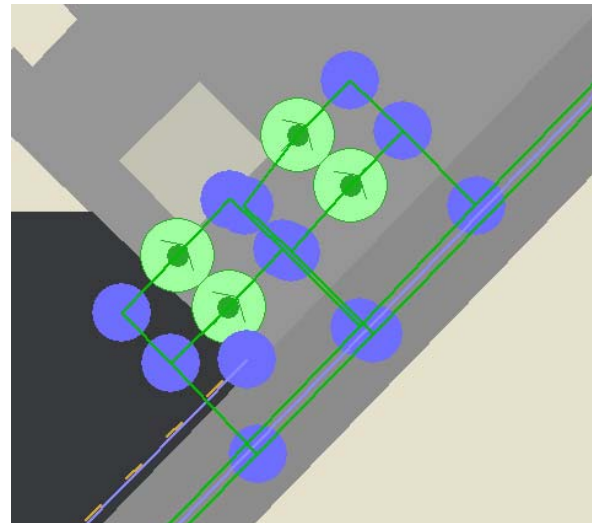
*The finished complex intersection*

ensure that your closed taxiways are dead-ended and not connected to your active taxiway system.

The method I found works best to minimize nodes is a combination of dead-end taxiways plus an apron polygon to shape the intersection before drawing the desired lines. Of course, this dead-ending of taxiways, by itself, is only useful if the taxiways are passive (such as in the base network of my dual-pipeline "drive-through" parking scheme). If AI must travel on the taxiways, you will have to add even more (invisible) links to reconnect the taxiways.

Finally, according to several forum posts, FS9 will use closed taxiways to route AI despite claims to the contrary. Therefore, you should

Parking and Parking Connectors - In order for “drive-through” parking to work, each parking spot must have its entry side connected to the arrivals pipeline (or base network) and its exit side connected to the departures network. It’s unlikely that you’ll be able to connect each parking spot directly to the pipelines, so you’ll want to find a way to link the arrivals side of several parking spots together and then connect this link to the arrivals pipeline (or base network), with a similar arrangement for departures. As noted earlier, don’t try to make the AI turn too sharply when entering or leaving a parking spot, otherwise arriving aircraft may stop before reaching the parking location and departing aircraft will push-back (but they will actually depart on the departures pipeline.)



*Segregated arrivals and departures parking networks*

The final parking link leading into each parking spot should be aligned to the parking spot orientation so that the aircraft will be able to turn to the correct orientation. The greater the angle of turn, the further back must be this final link. Any parking spots that have a centerline leading to them must have a 0-width apron link as the first parking connector on the exit side. This will terminate the centerline at the crossbar where the aircraft is to stop. The width of the final link into the parking spot should be equal to the desired width of the “stop” crossbar.

## “DRIVE-THROUGH” PARKING

Once the airport configuration is finalized, it’s time to draw the pipelines. The big question is: “One?” or “Two?” If you’ve not modified the airport to avoid fillets and eliminate “spider-webs”, you’ll probably find a single pipeline to be adequate. On the other hand, if you have significantly modified the airport, a dual pipeline approach is likely to be easier to implement. If in doubt, I suggest you start with dual pipelines. (It’s easier to revert to a single pipeline than to go the other way.)

But, before moving on, a word about conventions. While you may use any scheme you like, I use “blue” taxiway links for all physical taxiways and taxiway markings and “green” apron route links for pipelines and parking connectors. (Apron route links have the advantage that, unlike taxiway links, they have no surface structure. Hence, they can’t interfere with the visual presentation of the base network taxiways, which they overlay. As well, since there’s no surface structure to draw, using apron routes for pipelines should minimize any FPS “hit”.) Centerlines on these links are enabled only in certain parking areas – or for troubleshooting. For visual clarity, I offset the pipeline nodes and links laterally from the visible taxiway centerlines by a small amount, about 10’ (3 m.). The AI will travel along this offset path but, unless you have a very narrow taxiway, this shouldn’t be too distracting. You may use a smaller offset, but be cautious about getting your nodes too close to each other. On taxiways that parallel runways, I

draw the departure links on the side of the taxiway centerlines further away from the runway. This minimizes the number of times departure and arrival links cross. If the taxiway does not parallel a runway, experiment to determine which side is better.

In dual network configurations, while both pipelines are constructed of green apron routes, they are always separated (in the AFCAD 2.21 display) by the blue base network links. By consistently keeping the departure pipeline on far side of the base taxiway links from the runway and the arrivals pipeline on the near side, it will be easy to recognize which is which.

Finally, make a backup of your AFCAD file and put it somewhere where it won't accidentally get overwritten or deleted. Also, make frequent back-ups of your work in progress. Once you exhaust AFCAD 2.21's undo capability, going back is not easy.

Single Pipeline Configuration - The first step in implementing pipelining is to prepare your airport. For single pipeline implementations, this is limited to providing an extra node at the end of each runway, separated in each case from the nearest other runway node by a "black" runway link. (AI will not traverse "black" links if there is any other available route to the destination.) This may involve minor modifications of the runway end taxiway to make room for the extra node.

Starting at those runway-end nodes and following the base network taxiway, draw a network of nodes and apron route ("green") links without centerlines towards the parking areas. The width of these links is arbitrary. But, as noted earlier, in order for FS9 to detect an AI aircraft on a taxiway link or apron route, the link must be at least 20' wide.

While not essential for AI operation, to facilitate unambiguous ATC taxi instructions, each link in the pipeline should be designated with the correct taxiway name.

Add a hold-short note, drawn as previously described, for each departure position, not more than 225' (68 m.) from the runway. AFCAD 2.21's "H" function will allow you to confirm that your hold-short nodes are within the prescribed distance.

Since FS9 AI always depart from the outermost node of the runway, there is no need to connect, and the overall scheme relies on not connecting, the departures pipeline to any intermediate points on the runway.

Connect the departures network to the outbound side of all the parking spots. Lastly, connect the inbound side of your parking spots to the base network.

That's all there is to it. "Fire-up" some AI and check that everything works as you intended.

Dual-Pipeline Configuration - If the base taxiway network is not to be used for AI, as in a dual pipeline configuration, it is necessary to disconnect all taxiways from the runways. So, "dead-end" all the taxiways. This, of course, will disconnect the visible taxiway centerline from where it joins the runway, but don't worry about that until later.

In keeping with using the base network for visual purposes, leave the hold-short nodes in place.

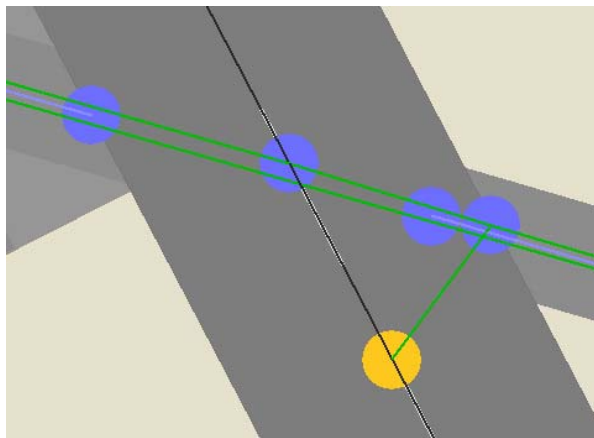
On the runways, the “black” runway taxi links running along the centerline (I’ll call it the “runway spline”) which previously interconnected taxiways and runways will now be just a series of links connecting nothing but the two ends of the runway. You may be inclined to delete all the intermediate links and nodes, replacing them with a single link that runs the length of each runway. However, there are several posts in the AFCAD forum at ProjectAI that suggest deleting nodes may lead to positioning errors in various aspects of your airport. That is not a “good thing”. Consequently, you should leave these nodes in place for now. (Most, if not all, will be used later.)

Finally, disconnect the taxiways from the parking spots as well. Your previously integrated airport now comprises three independent and disconnected elements: runways, dead-ended taxiways and parking.

Next, lay out the departures network as set out in the previous section – but leave the configuration the hold-short nodes ‘till later, doing both the arrivals and departures nodes at the same time.

The first step in constructing the arrivals network is to add a normal node (or ensure one exists) on the runway spline at each runway/taxiway intersection where traffic is to exit, and at the ends of the runway just inboard of the departure nodes. (Since the links between these departure and arrival nodes at the end of the runways are “black”, AI won’t transit them.) If you didn’t delete the old nodes when preparing the base network, just reuse them. Add any further nodes you need or delete any surplus ones.

After preparing the runways, construct a second pipeline, for arrivals, in a similar manner to the departure pipeline – this time keeping the arrivals nodes and links on the runway-side of the base network taxiway centerlines. Hold-short nodes must also be included in the arrivals network.

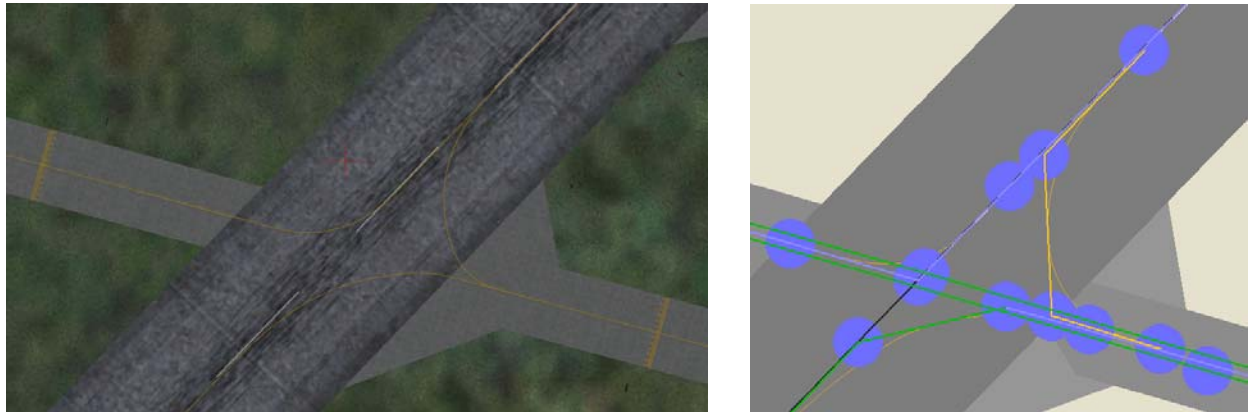


*Taxiway turning link*

(AI stop just beyond the first hold-short node they encounter after exiting the runway to wait for taxi-to-parking clearance. If you don’t provide dedicated hold-short nodes at each runway entrance to the arrivals pipeline, the AI will taxi to parking, or until it encounters another hold-short node, without receiving clearance.)

At some intersections on the arrivals network, and at taxiway intersections where the angle is very large ( $>90^\circ$ ), you may need to add a “turning link” where aircraft radius-of-turn takes the AI off or very close to the taxiway edge when turning off the runway.

You should now have two separate pipelines closely paralleling each other except, of course, at intermediate runway entrances/exits. The departures pipeline provides a path from the outbound side of the parking pipeline to the runway ends. The arrivals pipeline provides a path from each intermediate runway/taxiway connection to the inbound side of the parking spots. The two networks should only connect together through the parking spots and at the runway ends, in the latter case, tied together with a “black” runway taxi link.



*The finished intersection*

In dead-ending the visible taxiways earlier, the taxiway centerline, if there was one, was disconnected from the runway centerline. It can easily be replaced using decorative taxiway links as previously described. Because the taxiway will have been cut-off close to the runway, it may be necessary, depending on the desired radius of turn, to place a short link without a centerline at the end of the taxiway so as to move the end of the visible taxiway centerline back away from the intersection. .

Controlling Which Runway Exit is Used - Purely by accident, I discovered that, while AI generally uses the first runway exit encountered after slowing down, at times FS9 can be “choosy”.

FS9 appears to have an aversion to runway exits where the aircraft must turn through more than 90°. Indeed, AI will pass up such an exit to go further down the runway to reach one which has a lesser angle. I have not done any significant amount of experimentation on this matter. Rather, I mention it here simply to make you aware of the possibilities. For example, if you have a situation where you want the AI from one direction not to use an exit, placing the link between the taxiway and the runway so as to increase the angle of intersection from that direction may force AI to seek another exit. Of course, this may enable exits by traffic from the other direction, so how far down the runway exit is located will also be a factor in how you design your runway exit strategy.

Special Taxi Routes - If the airport you are modeling has different taxi routes for different classes of aircraft, you may also be able to implement this using pipelines. At CYYJ, a hanger is located quite close to one of the taxiways, so close that it’s not safe for wide-bodied aircraft to use that taxiway. Instead, wide bodied aircraft bypass this narrow taxiway by traveling part way

to the terminal along a lightly-used runway. To replicate this, I created two secondary pipelines, one for arrivals, the other for departures, extending from the point where the “wide-bodies” exit the taxiway in favor of the runway, along the runway, and then to the parking spots for wide-bodied aircraft at the terminal (which are not connected to the main arrival and departure pipelines). Consequently, the wide-bodies’ only paths between their parking and the point where they diverge/converge with the main taxiway network are these secondary pipelines.

### A FINAL THOUGHT

While pipelining appears to offer a general solution for implementing “drive-through” parking, it is not the only solution.

FS 9 routes AI via the shortest available route to its destination. FS9 doesn’t just count links, it actually calculates each candidate-link length and, hence, the actual total length of each prospective taxi path – picking the shortest total path. Some have implemented a limited “drive-through” parking arrangement (e.g., for a specific parking area) based on this “shortest route logic”. However, if arriving and departing AI share any portion of a taxiway, every taxiway that eventually links to that portion becomes a potential path for both arriving and departing AI. So, it’s unlikely you will be able successfully to implement a “drive-through” parking scheme using “shortest route logic” that will work across the whole airport with any runway being active.

But, where you can find a way to:

- segregate arriving and departing AI (in the vicinity of a take-off-only or landing-only runway, any given taxiway will carry either arriving or departing traffic, but not both; as well, by omitting strategic taxiway links such that AI is prevented from using certain portions of a taxiway, other parts of the taxiway may be made effectively one-way); or
- in a localized area, guarantee that the path from parking to takeoff will be shorter than the path for arrivals to parking (as would be the case if parking was located very close to the take-off end of the active runway);

it may be possible to implement “drive-through” parking for a portion of your airport without using pipelines. But, this may also require artificial constraints on, for example, the choice of active runway(s) and, hence, loss of realism – and will certainly call for a good deal of inventiveness on your part.

The illustrations in this tutorial offer a rather simplified view of some of the steps involved. You are encouraged to use AFCAD 2.21 to explore the two “.bgl” files included in the folder where you found this tutorial. One (“AF2\_CYYJ\_(2).bgl”) is a single-pipeline implementation for the almost finished CYYJ (2006). The other (“AF2\_CYYJ\_(3).bgl”) is the final AFCAD for CYYJ (2006). From these two files, you should gain a much better appreciation as to what’s involved in implementing “drive-through” parking.

Revisiting Lee Swordy’s comment in the Help feature of AFCAD 2.21 “because of limitations with the AI, it [drive-through parking] rarely results in the kind of behaviour one would hope”, I have to wonder if maybe he was referring to node proximity difficulties and problems with AI receiving taxi-to-parking clearance. If you think about it, pipelines (one or two) ensure the

correct operation of “drive-through” parking because any taxiways accessible to arriving AI lead only to the inbound side of the parking spots and all routes from the outbound side of the parking spots lead only to AI take-off positions. It can’t fail (so long as “node-proximity” doesn’t “get you”)!

Good luck with your airport,

Don Grovestine  
[dgrovestine@shaw.ca](mailto:dgrovestine@shaw.ca)  
May 1, 2006