AGUACLARA: CREATING AN AUTOMATED DESIGN TOOL

A Thesis

Presented to the Faculty of the Graduate School

of Cornell University

In Partial Fulfillment of the Requirements for the Degree of

Master of Engineering

by

Anastasia Rudenko

December 2008

© 2008

ABSTRACT

The following paper provides an outline of the automated design tool used by the AguaClara project team. The automated design tool uses commands coded in Mathcad to produce an AutoCad rendition of water treatment plants based on user defined parameters.

BIOGRAPHICAL SKETCH

Anastasia Rudenko completed her undergraduate degree in Biological and Environmental Engineering at Cornell University in May 2008. While an undergraduate she worked on the Solar Decathlon project team. She started work on her Masters of Engineering as an early admission student in January 2008. For her Master's design project Anastasia participated in the AguaClara project team. This January she will be working as a project engineer with Stearns & Wheler in Raleigh, North Carolina.

I dedicate this paper to my family.

ACKNOWLEDGMENTS

I would like to thank Professor Michael Timmons, my undergraduate advisor and Master's chair advisor, for all the guidance that he has given me over the last four and a half years. I would also like to thank Professor Monroe Weber-Shirk for guiding my work on the AguaClara project team.

APPENDICIES

CHAPTER 1

AguaClara Project Team

The AguaClara Project

The Millennium Development Goals (MDG) outlines an ambition to reduce child mortality and ensure environmental sustainability. A prevalent source of both problems is contaminated water sources, which cause preventable diseases such as diarrhea. Such illnesses are responsible for the deaths of approximately 5000 children every day. Currently it is estimated that over one million people worldwide do not have access to clean drinking water. The AguaClara project focuses on developing technologies which can be implemented to help fulfill these goals.

Many of the technologies that exist for water treatment in developing countries are difficult to practically implement. Slow sand filters (SSF) are unable to handle high turbidity events making them unusable for many communities and multiple stage filtration (FIME) requires a large plant footprint which leads to high construction and operation costs. Hydraulic powered conventional water treatment systems (HPCWT) are problematic due to complications related to backwashing deep filtration boxes without using electricity to run a pumping system. Point of use (POU) systems are often used in rural communities where distribution systems are non-existent and would be expensive to build, however it is impossible to monitor water quality in these systems and contamination can occur easily with improper use.

In communities where piping distribution systems exist, municipal water treatment systems offer several advantages, including economies of scale as well as requiring fewer maintenance personnel for operation. AguaClara has developed a water treatment technology that is competitive with POU devices on a POU basis for cities with pre-existing distribution systems. Capital costs for the project are less than $20 per person and operating costs are about $2 per day. The project team is currently

working with cities of under 50,000 residents that already have an established piping distribution system, and has provided designs for water treatment plants for four Honduran communities that serve a combined 13,000 people. This semester two other plants are in the design and construction phase. All of the plants are built from locally bought materials and require no electricity for operation.

AguaClara Technology

One of the constraints for AguaClara is to design a system that can be used in areas with intermittent electricity sources. The team is working to optimize a hydraulic flocculation/sedimentation process to remove particles in the raw water before chlorine disinfection.

Removing as many particles from raw water as possible before disinfection is very important when trying to minimize chemical costs. Chlorine reacts with organics in the water and forms carcinogenic compounds. Additionally, chlorine is "used up" in these reactions and is unavailable for disinfection. Water being treated by chlorine disinfection should have a turbidity of less than ten NTU to maintain an adequate chlorine residual. Removing particles through preliminary flocculation/sedimentation treatment minimizes both health risks and the amount of chlorine that must be added to the system.

Raw water first enters the plant through pre-existing transmission lines into an entrance tank then leaves through riser pipes. The riser pipes are designed to provide a linear relationship between the depth of water in the entrance tank and flow rate. Water leaving the entrance tank is mixed with alum, a coagulant.

Alum is introduced into the system through a flow control module (FCM) where a float valve in the FCM ensures a constant head in the system. Alum dosing is controlled by manually adjusting the height of the exit tubing. By determining head

loss, the elevation of the exit tubing controls the alum flow rate. Operators are trained to adjust the alum flow rate as the turbidity of the incoming water changes and adjustments are made based on the recommendations of charts that translate influent turbidity into a required tube inlet height.

The alum and raw water go through a rapid mix process where they are forced through a series of pipe elbows. The water then enters a vertical flocculator where it is forced through a series of channels created by baffles made of plastic roofing, a local building material. Via mixing, alum is dispersed throughout the water stream and coats suspended particles. The coagulant decreases the net charge on suspended dirt particles making it more likely for them to stick together as they collide. The resulting clumps of particles that form are called flocs. The AguaClara research team is currently concentrating on optimizing the spacing of the baffles to produce large flocs.

After the water leaves the flocculator it enters into the bottom of the sedimentation tank through entrance pipes and flows up through an array of lamella. Flocs settle out of the slowly rising water and slide down the lamella to the bottom of the tank. Clean water is collected from the top of the tank through manifolds. Research is being conducted into utilizing the sludge buildup at the bottom of the tank in a 'sludge blanket' which would act as a diffuse filter for incoming water. Preliminary results have yielded a water turbidity of less than one NTU when the sludge blanket is used. Periodically the tanks must be drained to clean out sludge deposition. Even with the sludge blanket the combined flocculation/sedimentation process is capable of producing water that is less than 5 NTU.

AguaClara Technology

One of the design constraints for AguaClara is to design a system that can be implemented in areas with intermittent electricity sources. The team is working to optimize a hydraulic flocculation/sedimentation process to remove particles in the raw

water before disinfection through chlorination. Chlorine reacts with organics in the water and forms carcinogenic compounds. In addition to this health hazard chlorine is "used up" in these reactions and thus unavailable for disinfection. Water being treated by chlorine disinfection should have a turbidity of less than ten NTU to maintain an adequate chlorine residual. Removing particles through preliminary flocculation/sedimentation treatment minimizes both health risks and the amount of chlorine that must be added to the system (reducing costs.)

Water first enters the plant through pre-existing transmission lines to an entrance tank. Water leaves the tank through riser pipes. The riser pipes are designed to provide a linear relationship between the depth of water in the entrance tank and flow rate. Water leaving the entrance tank is mixed with alum, a coagulant. Alum is introduced into the system through a flow control module (FCM.) A float valve in the FCM ensures a constant head in the system. Alum dosing is controlled by manually adjusting the height of the exit tubing. By determining head loss, the elevation of the exit tubing controls the alum flow rate. Operators are trained to adjust the alum flow rate as the turbidity of the incoming water changes. Adjustments are made based on the recommendations of charts that translate influent turbidity into a required tube height.

The alum and raw water go through a rapid mix process as they are forced through a series of pipe elbows. The water then enters a vertical flocculator where it is forced through a series of channels created by 'baffles' made of plastic roofing, a local building material. Via mixing alum is dispersed throughout the water stream and coats suspended particles. The coagulant decreases the net charge on suspended dirt particles making it more likely for them to stick together as they collide. The resulting clumps of particles that form are called flocs. The AguaClara research team is currently concentrating on optimizing the spacing of the baffles to produce large flocs.

After the water leaves the flocculator it enters into the bottom of the sedimentation tank through entrance pipes then flows up through an array of lamella. Flocs settle out of the slowly rising water and slide down the lamella to the bottom of the tank. Clean water is collected from the top of the tank through manifolds. Research is being conducted into utilizing the sludge buildup at the bottom of the tank as a diffuse filter for incoming water. Preliminary results have yielded a water turbidity of less than one NTU when a 'sludge blanket' is used. Periodically the tanks must be drained to clean out sludge deposition.

Even without the sludge blanket the combined flocculation/sedimentation process is capable of producing water that is less than 5 NTU. The water enters an exit chamber where chlorine is added to kill remaining pathogens. Chlorine is dosed using a flow control module similar to the alum dosing system in the entrance tank. Water is piped to into people's homes from the distribution tank.

Creating an Automated Design Tool

The AguaClara project team operates under the principles of open source engineering in which accumulated knowledge is openly shared and accessible to anybody with an interest in the project. In adhering to this philosophy, it is important to develop a well-documented design algorithm that can be used to produce renditions of AguaClara plants.

Currently municipal water treatment systems take a long time to complete because each system is designed for specific locations. In response to this concern the AguaClara team is designing an automated tool that can be used to streamline the design process, allowing treatment plants to be produced more rapidly. Through the automated design tool, engineers can enter parameters, such as plant flow rates, to generate an AutoCad rendition of a plant designed to specifications.

AguaClara has developed a method to code AutoCad commands into the engineering program Mathcad. The Mathcad script can be outputted to a notepad document which can easily be entered into the AutoCad command window to produce a rendition of the treatment plant. Renderings are then emailed to interested engineers. Eventually an even more user-friendly interface will be developed utilizing LabView software.

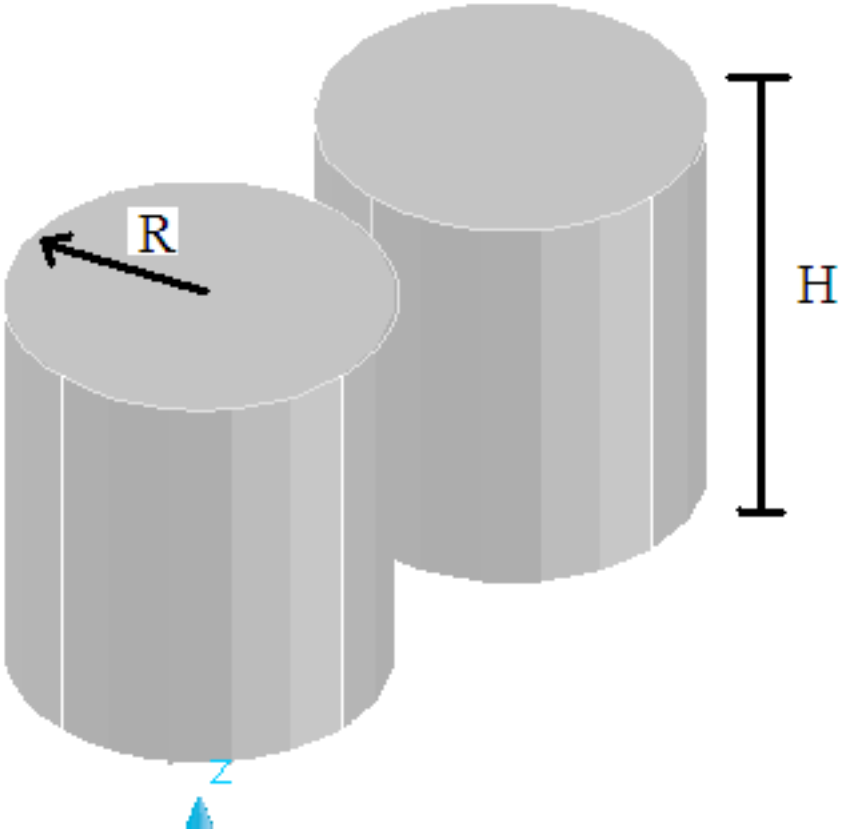
Each piece of the treatment system is encoded in a separate MathCad file. All the files are compiled in a master program that produces the final rendition of the plant. Last semester I worked with the design team to develop stand-alone scripts for the plant channel, sloped sedimentation wall, lamella and baffles. After the scripts were completed, the latter part of the semester was spent integrating the baffle script into the flocculation tank program, and the channel and slope scripts into the sedimentation tank program. The latest version of the scripts can be found on the AguaClara SourceForge, a file sharing program.

AutoCad Algorithm Documentation

Last semester we completed most of the separate algorithms for the individual pieces of the plant. However these codes are constantly being updated as new research innovations are incorporated into our design. With a rapid turnover in team members, it is important to provide written documentation for how the algorithms were written so they can be easily modified. This semester I focused on compiling a line by line guide for the Mathcad code we had compiled. The program guidelines are located on the AguaClara website and accessible to all interested parties.

This page last changed on Dec 11, 2008 by ar329.

General Program Information



Northwest view

Input Definitions

Inputs Needed to Call the Chemical Stock Tank Barrels Function

- origin - a 3*1 matrix with user defined x,y,z positions corresponding to origin. The origin is located at the top right corner of the platform.
- disp - displacement between the edge of the drum containing the chemical stock solution (sometimes also refered to as the chemical stock barrel) and the edge of the platform.
- thick - specifies the thickness of the platform.
- walkway - the width of a walkway space on the platform, so that the plant operater can walk on the platform next to the chemical stock drums/barrels
- R - chemical stock drum radius/ chemical stock barrel radius
- H - chemical stock drum radius/ chemical stock barrel height

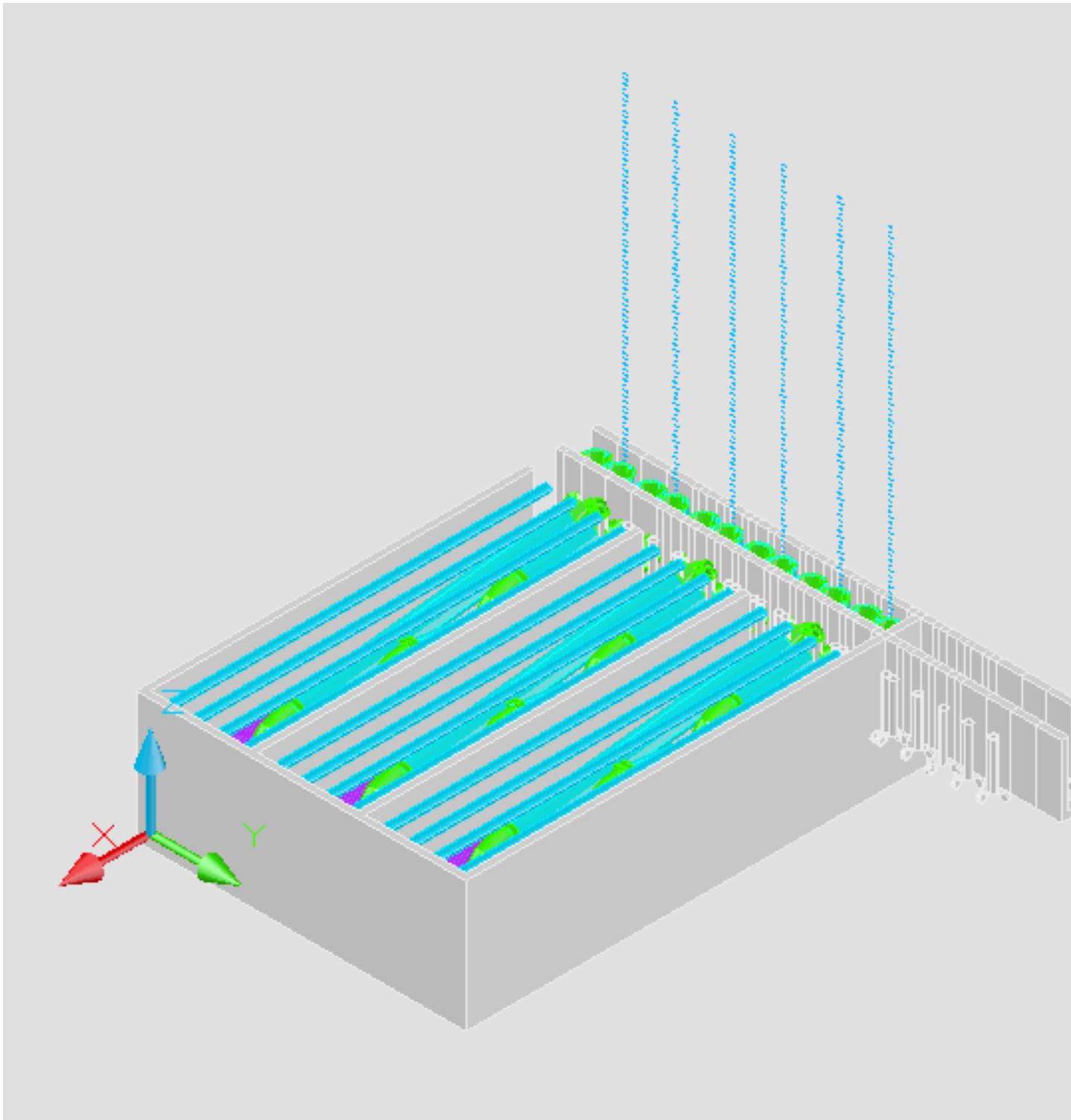
Inputs Defined within the Chemical Stock Tank Barrels Function

$platform_{origin}$ =

This page last changed on Nov 25, 2008 by ar329.

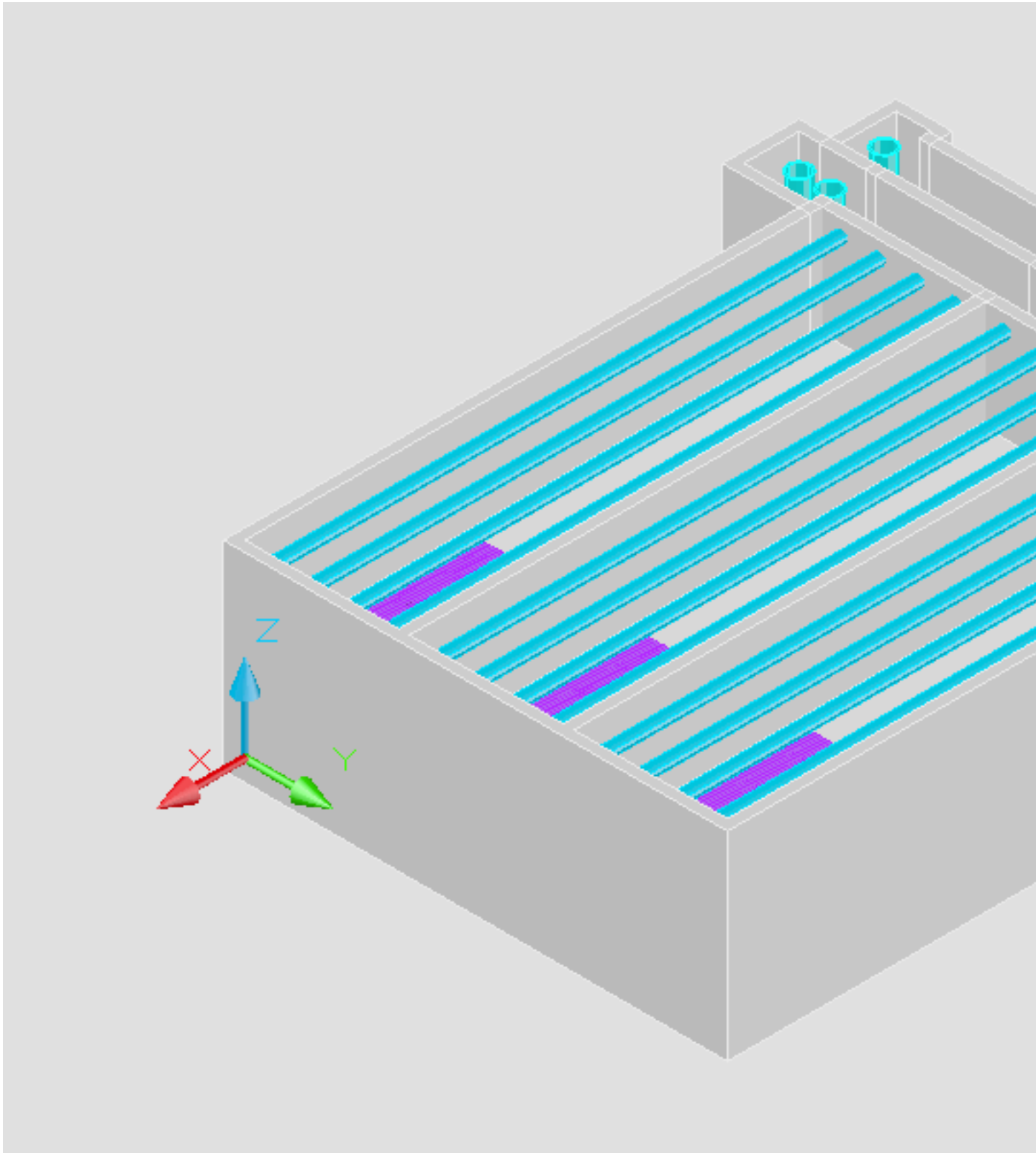
Automated Design AutoCAD Sedimentation Tank Program

Sedimentation Tank Layouts



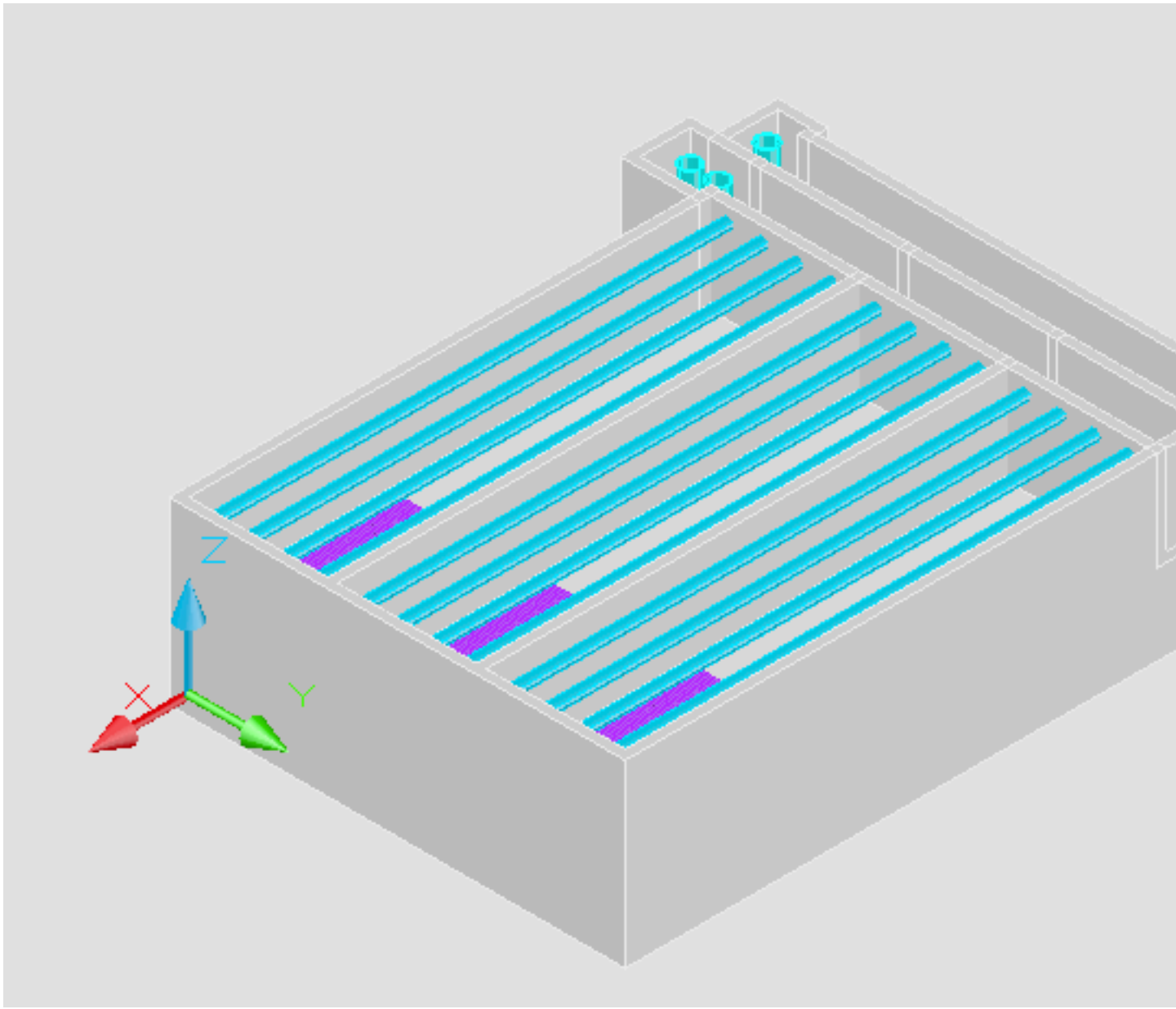
Northeast Isometric View

Sedimentation Tank Layout 1(needs pictures and to be reviewed)



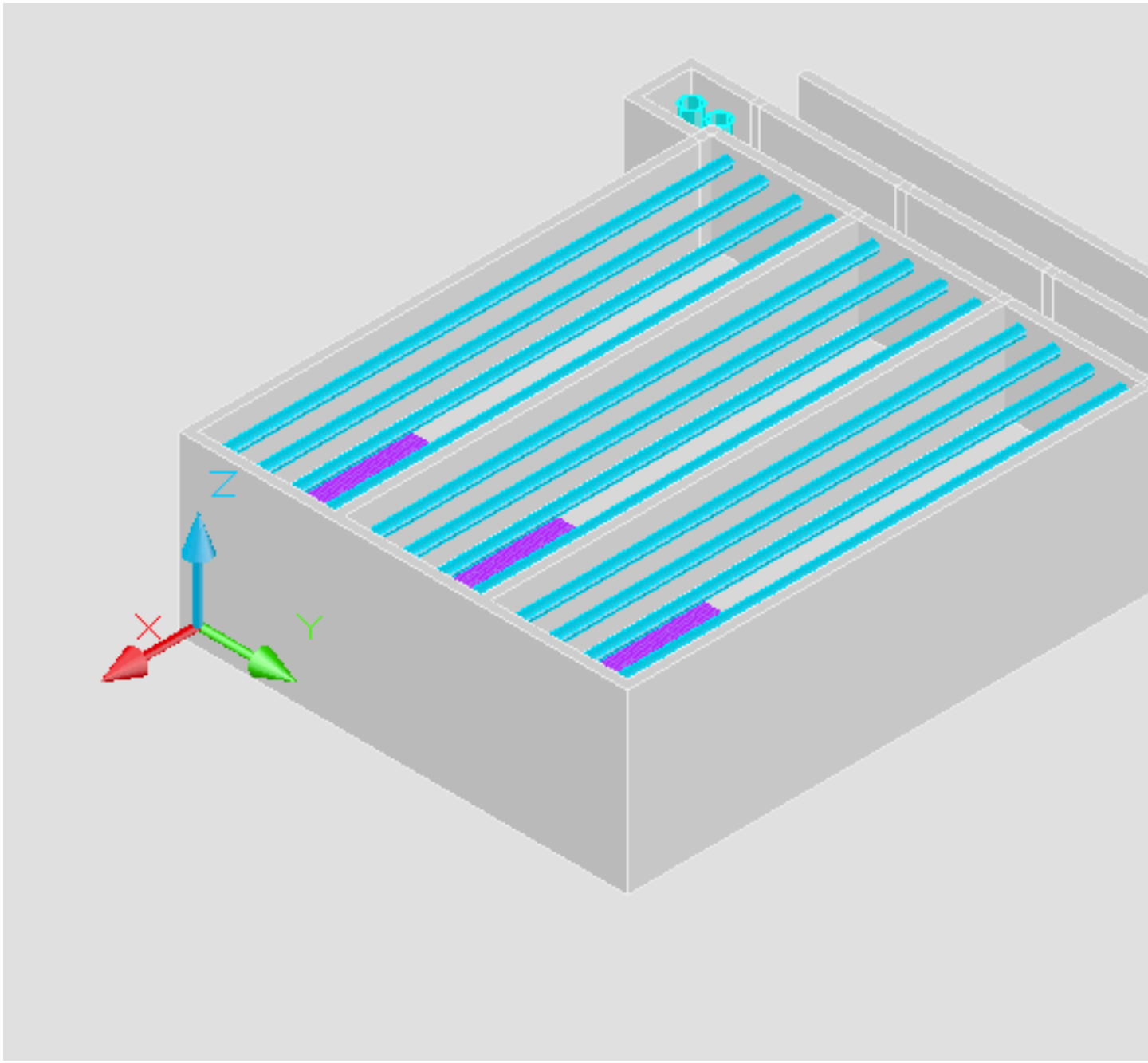
Northeast Isometric View

Sedimentation Tank Layout 2(needs pictures and to be reviewed)



Northeast Isometric View

Sedimentation Tank Layout 3(needs pictures and to be reviewed)



Northeast Isometric View

Sedimentation Tank Layout 4(needs pictures and to be reviewed)

Program Pieces

Sedimentation Tank Inputs
Sedimentation Tank Origin Points (needs pictures)
Sedimentation Tank Pipe Lengths (ready for initial review - Anastasia)
Sedimentation Tank Pipe Rotation Angles (ready for initial review - Anastasia)
Sedimentation Tank Elbow Origins (needs more info and pictures)
Sedimentation Tank Sedtank Scripts(in progress)

This page last changed on Nov 11, 2008 by ar329.

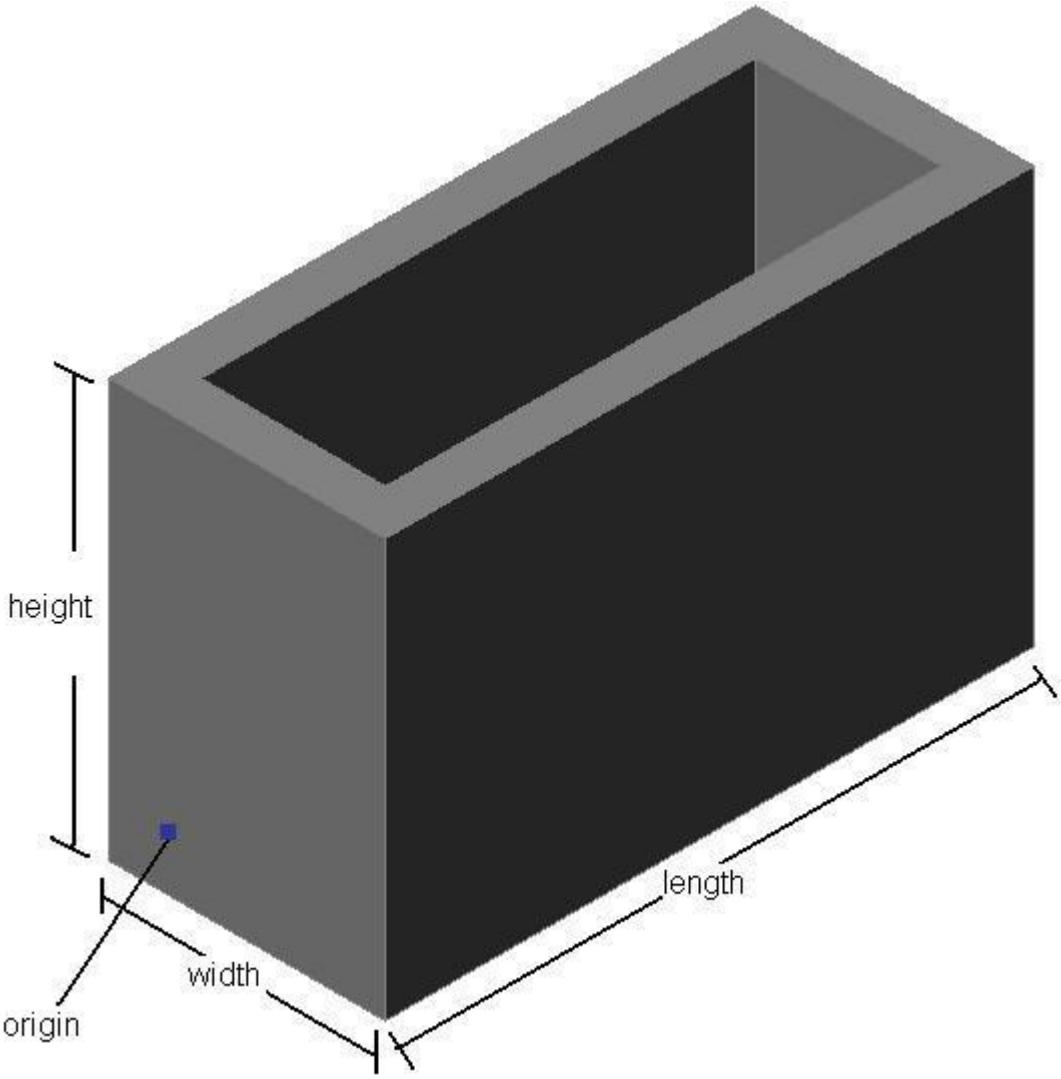
Tank Drawing Script

viewtops - sets the workspace so that the user is viewing the top of the object

viewtops <- viewtop1

layer1 - $Layer_{new}$ creates a new grey layer "tank."

layer1 <- $layer_{new}$("tank", grey)



Northeast Isometric View

tank1 - Calls the Tank Program to create a tank based on three inputs.

Note: p1 is a dummy variable used only in the program help section to designate the matrix below.

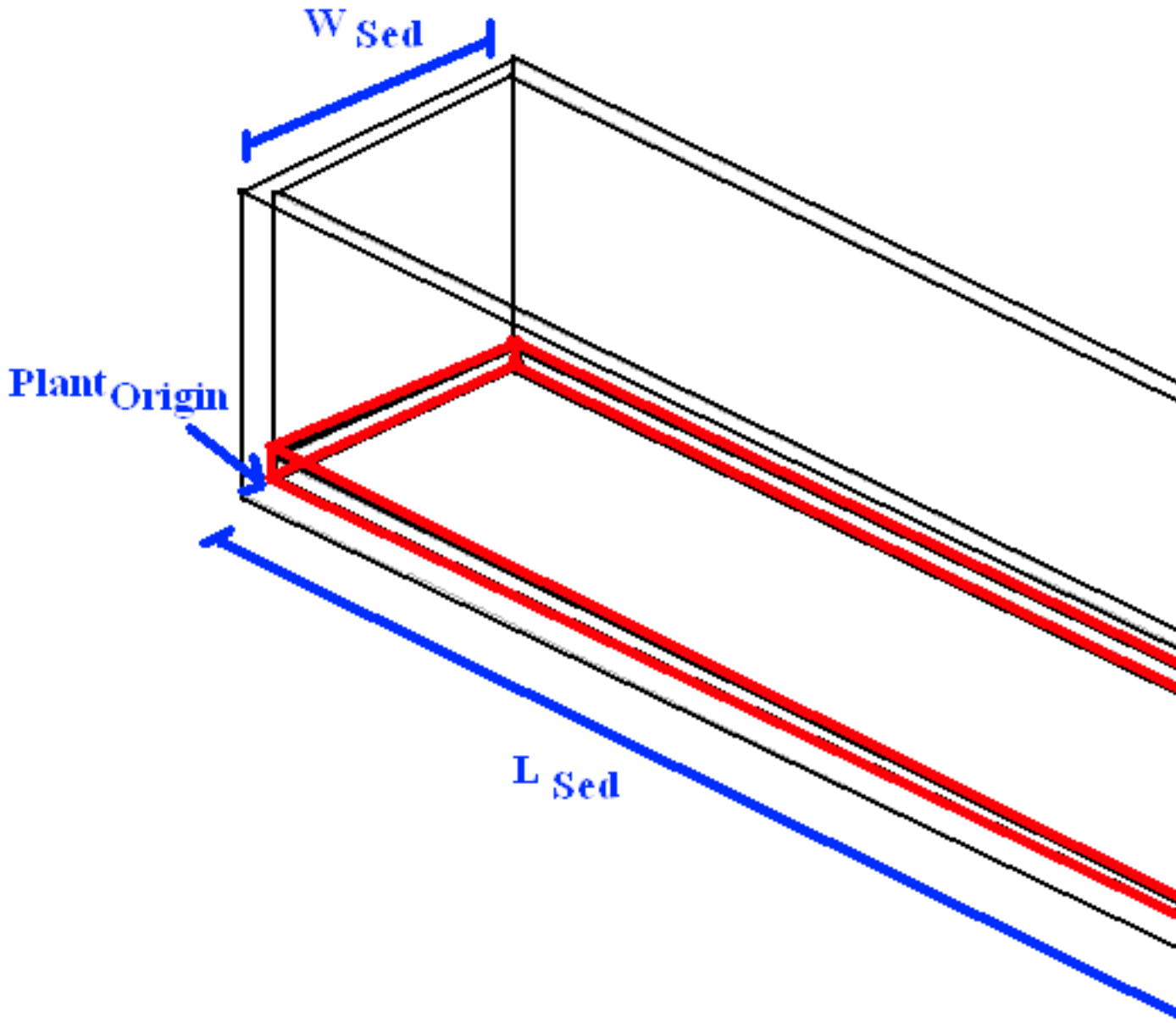
tank1 <- Tank($Plant_{Origin}$, p1,$T_{PlantWall}$)

$Plant_{Origin}$ =

p1 =

- x: L_{Sed}
- y: W_{Sed}
- z: H_{Sed}

$T_{PlantWall}$ = 0.15m



---

boxt - Creates a box based on two points.

boxt <- box($Plant_{Origin}$,$sedbottombox_{dim}$)

$Plant_{Origin}$ =

$sedbottombox_{dim}$ =

- x: -L_{Sed}
- y: W_{Sed}
- z: outerdiameter($ND_{SedSludge}$)

$ND_{SedSludge}$ = Sludge pipe diameter in sedimentation tank
uniont - $Union_{allA}$ selects all the objects in the workspace and unions them into a single object.

uniont <- $union_{allA}$

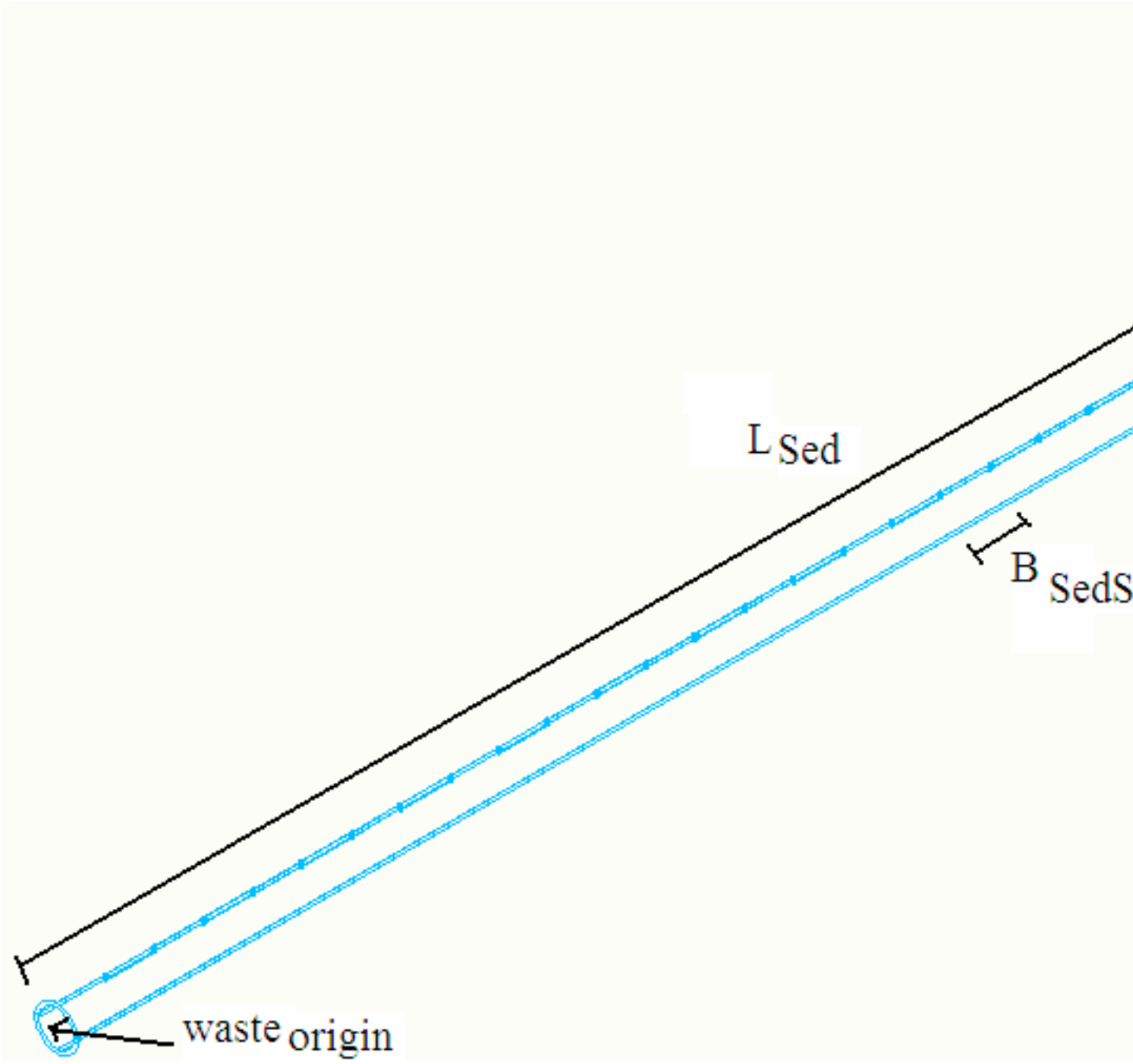
layerset - $Layer_{set}$ selects the layer that the user is currently working in.

layerset <- $layer_{set}$("0")

layerfreeze1 - $Layer_{freeze}$ locks the selected layer so that it cannot be edited.

layerfreeze1 <- $layer_{freeze}$("tank")

This page last changed on Dec 17, 2008 by ar329.

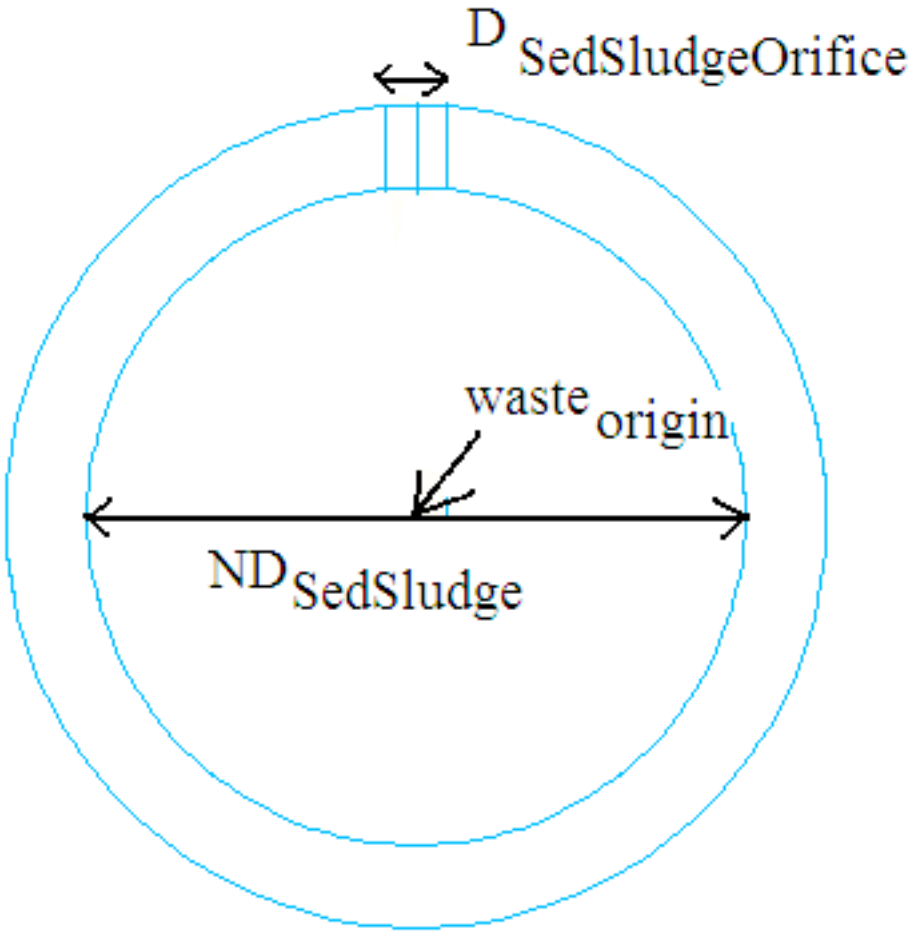


NorthEast Isometric View

Sludge Pipe Drawing Script

layer6 - Layer$_{new}$ creates a new blue3 layer "sludgepipe."

layer <- layer$_{new}$("sludgepipe",blue3)



Top View

sludgepipe1 - Calls the Sludge Pipe Program to create a sludge pipe.

sludgepipe1 <- Sludgepipe($waste_{origin}$,$ND_{SedSludge}$,L_{Sed},$D_{SedSludgeOrifice}$/2,$B_{SedSludgeOrifices}$,$EN_{PipeSpec}$)

$waste_{origin}$ =

- x: $tank_{origin0}$ - L_{Sed}
- y: $tank_{origin1}$ + W_{Sed}/2
- z: $tank_{origin2}$ + outerdiameter($ND_{SedSludge}$)/2

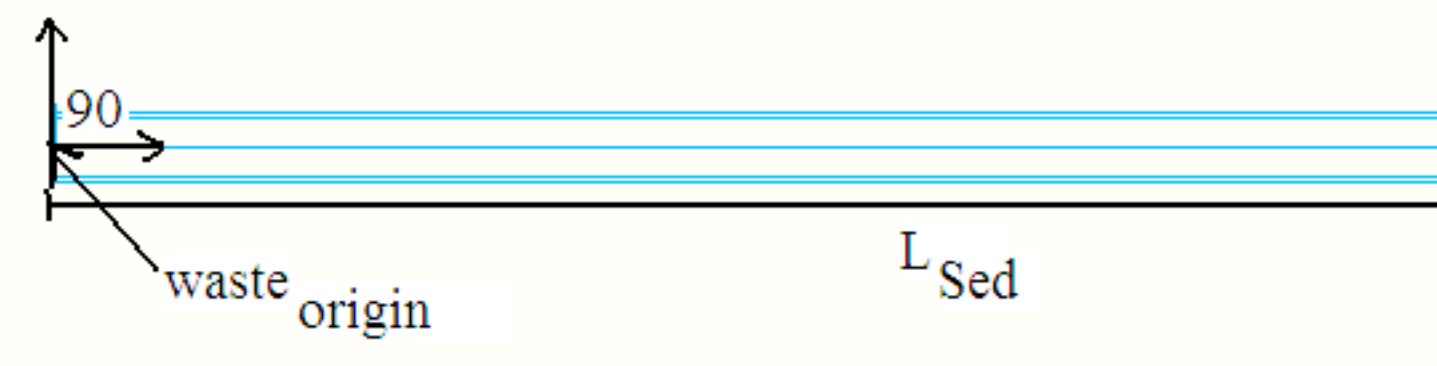
$ND_{SedSludge}$ - Sludge pipe diameter in sed tank.

L_{Sed} - Length of sed tank.

$D_{SedSludgeOrifice}$ - Diameter of orifices in sludge pipe.

$B_{SedSludgeOrifices}$ - Orifice spacing in sedimentation tank draining pipe.

$EN_{PipeSpec}$ - Enumerated type.



Top View

rotate20 - [Rotate$_{3d}$] turns the object based on a given axis and degree angle.

rotate20 <- rotate$_{3d}$(p1,waste$_{origin}$,"y",90)

Note: p1 is a dummy variable used only in the program help section to designate the matrix below.

p1 =

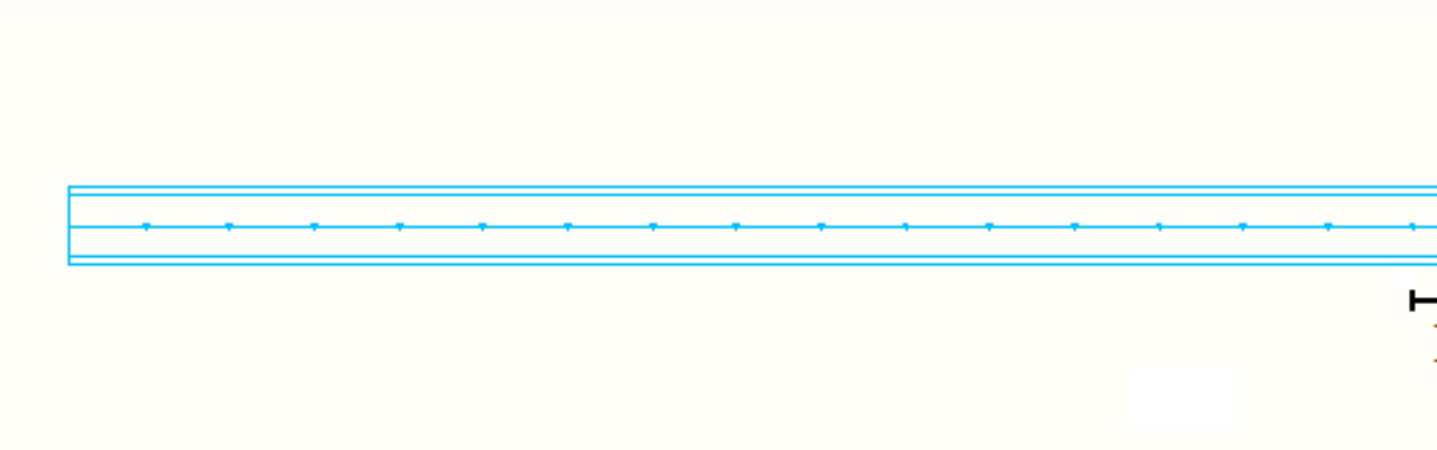
- x: waste$_{origin0}$ + innerD(ND$_{SedSludge}$,EN$_{PipeSpec}$)/2
- y: waste$_{origin1}$
- z: waste$_{origin2}$

waste$_{origin}$ =

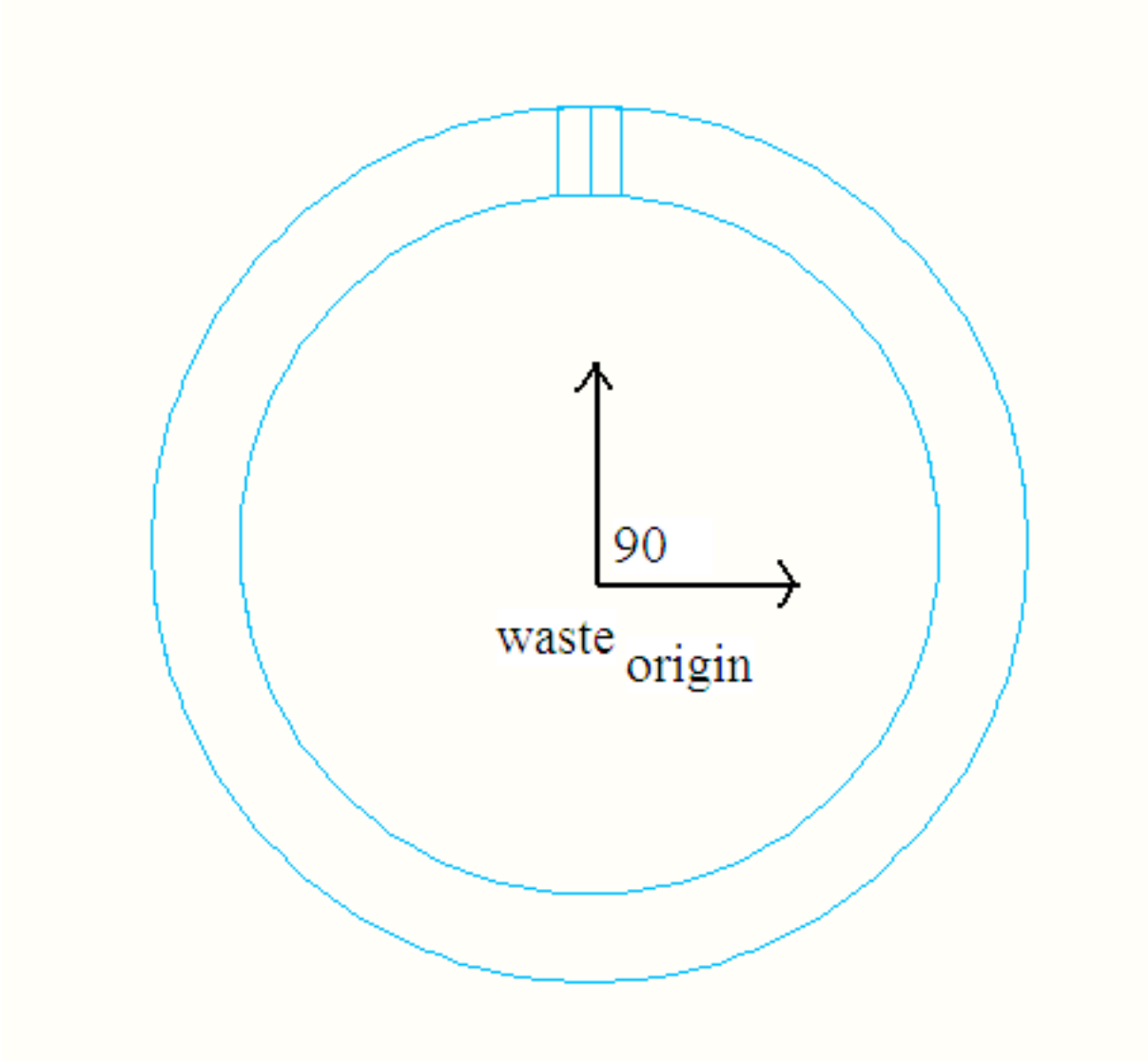
- x: tank$_{origin0}$ - L$_{Sed}$
- y: tank$_{origin1}$ + W$_{Sed}$/2
- z: tank$_{origin2}$ + outerdiameter(ND$_{SedSludge}$)/2

"y" - specifies axis that object will be rotated about.

90 - rotation angle



Top View



Right View

rotate21 - Rotate$_{3d}$ turns the object based on a given axis and degree angle.

rotate21 <- rotate$_{3d}$(p1,waste$_{origin}$,"x",90)

Note: p1 is a dummy variable used only in the program help section to designate the matrix below.

p1 =

- x: waste$_{origin0}$
- y: waste$_{origin1}$ + innerD(ND$_{SedSludge}$,EN$_{PipeSpec}$)/2
- z: waste$_{origin2}$

waste$_{origin}$ =

- x: tank$_{origin0}$ - L$_{Sed}$
- y: tank$_{origin1}$ + W$_{Sed}$/2
- z: tank$_{origin2}$ + outerdiameter(ND$_{SedSludge}$)/2

"x" - specifies axis that object will be rotated about.

90 - rotation angle
layerset - Layer$_{set}$ selects the layer "0".

layerset <- layer$_{set}$("0")

layerfreeze6 - Layer$_{freeze}$ locks the layer "sludgepipe" so that it cannot be edited.

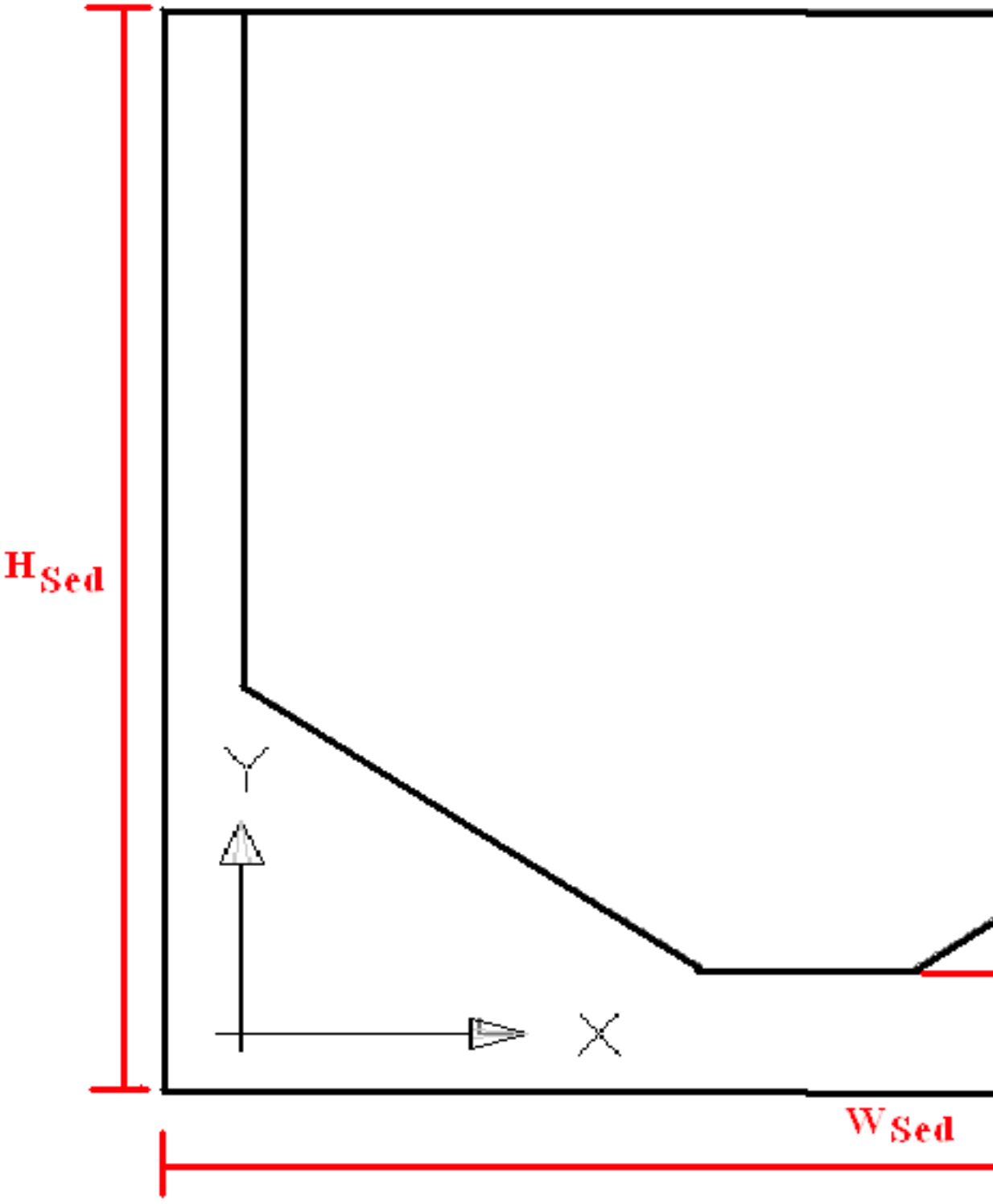
layerfreeze6 <- layer$_{freeze}$("sludgepipe")

This page last changed on Nov 20, 2008 by ar329.

Tank Slope Drawing Script

layerslopes - $Layer_{new}$ creates a new light grey layer, "slopes."

layerslopes <- $layer_{new}$("slopes",ltgrey)



Right View

slopes - Calls the Sedimentation Tank Slopes Program to draw the tank slopes based on three inputs.

slopes <- sedslope($SedSlopes_{Origin}$,p1,$AN_{SedBottom}$)

Note: p1 is a dummy variable used only in the program help section to designate the matrix below.

$SedSlopes_{Origin}$ =

- x: $Plant_{Origin0}$
- y: $Plant_{Origin1}$
- z: $Plant_{Origin2}$

p1 =

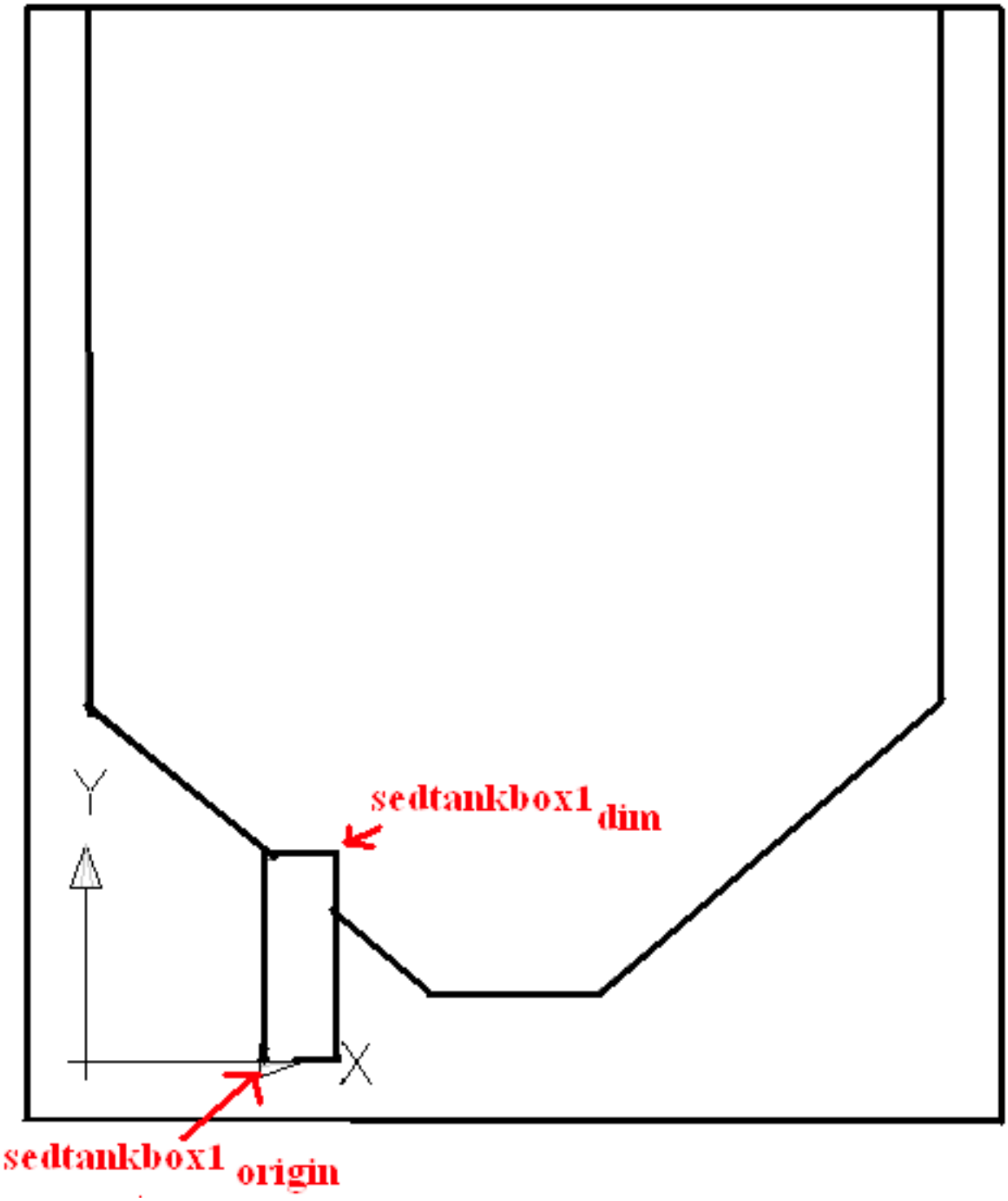
- x: L_{Sed}
- y: W_{Sed}
- z: H_{Sed}

$AN_{SedBottom}$ = Sedimentation tank bottom slope.
tankthaw - $Layer_{thaw}$ unlocks the "tank" layer so that edits can be made.

tankthaw <- $layer_{thaw}$("tank")

bigunion - $Union_{allA}$ selects all the objects in the workspace and unions them into a single object

bigunion <- $union_{allA}$



Right View

box1 - Creates a box based on two points.

box1 <- box($sedtankbox1_{origin}$, $sedtankbox1_{origin}$ + $sedtankbox1_{dim}$)

$sedtankbox1_{origin}$ =

- x: $Plant_{Origin0}$ - L_{Sed}
- y: $Plant_{Origin1}$ + W_{Sed}/2 - outerradius($ND_{SedSludge}$)

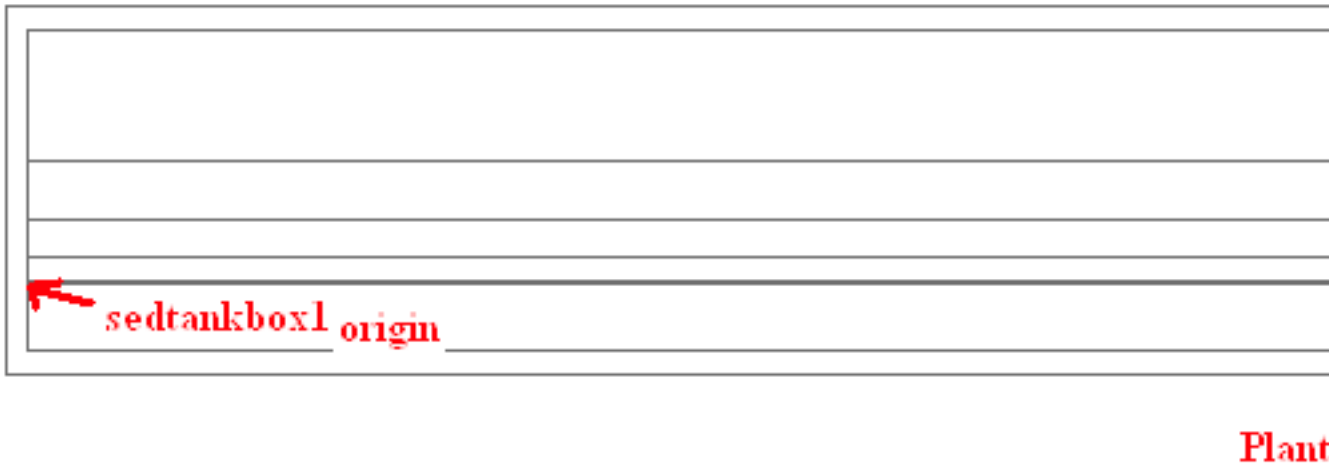
- z: $Plant_{Origin2}$

$sedtankbox1_{dim}$ =

- x: L_{Sed}
- y: outerdiameter($ND_{SedSludge}$)
- z: 3outerdiameter($ND_{SedSludge}$)

tankthaw - $Layer_{thaw}$ unlocks the layer "tank" so that it can be edited.

tankthaw <- $layer_{thaw}$("tank")



Top View

subtractbox - SubtractJ subtracts one object from the other based on four points.

subtractbox <- subtractJ($Plant_{Origin}$,p1,p2,$sedtankbox1_{origin}$)

Note: p1 and p2 are dummy variable used only in the program help section to designate the matrix below.

$Plant_{Origin}$ =

p1 =

- x: $Plant_{Origin0}$
- y: $Plant_{Origin1}$ + W_{Sed}
- z: $Plant_{Origin2}$

p2 =

- x: $Plant_{Origin0}$
- y: $Plant_{Origin1}$ - $T_{PlantWall}$

- z: $Plant_{Origin2}$

$sedtankbox1_{origin}$ =

- x: $Plant_{Origin0}$ - L_{Sed}
- y: $Plant_{Origin1}$ + $W_{Sed}/2$ - outerradius($ND_{SedSludge}$)
- z: $Plant_{Origin2}$

layerset - $Layer_{set}$ selects the layer "0".

layerset <- $layer_{set}$("0")

layerfreezeslope - $Layer_{freeze}$ locks the layer "slopes" so that it cannot be edited.

layerfreezeslope <- $layer_{freeze}$("slopes")

This page last changed on Nov 11, 2008 by ar329.

Open Channel Drawing Script

layerthawtank - $Layer_{thaw}$ unlocks the layer "slopes" so that it can be edited.

layerthawtank <- $layer_{thaw}$("slopes")

viewtop - Rotates the workspace so that the object is viewed from the top.

viewtop <- viewtop1

box1 - Creates a box based on two points.

box1 <- box($channelbox_{origin}$,$channelbox_{origin}$ + $channelbox_{dim}$)

$channelbox_{origin}$ =

- x: $tank_{origin0}$ - L_{Sed}
- y: $tank_{origin1}$
- z: $tank_{origin2}$ + H_{Sed} - $H_{Channel}$

$channelbox_{dim}$ =

- x: $W_{Channel}$
- y: $N_{SedTanks(W}$Sed~ + $T_{PlantWall}$)
- z: $H_{Channel}$

subtract1 - SubtractD subtracts one object from the other based on two points.

subtract <- subtractD($Plant_{Origin}$,$channelbox_{origin}$)

$Plant_{Origin}$ =

$channelbox_{origin}$ =

- x: $tank_{origin0}$ - L_{Sed}
- y: $tank_{origin1}$
- z: $tank_{origin2}$ + H_{Sed} - $H_{Channel}$

box2 - Creates a box based on two points.

box2 <- box($echannelbox_{origin}$,$echannelbox_{origin}$ + $echannelbox_{dim}$)

$echannel_{origin}$ =

- if layout1:
  - x: 0
  - y: 0
  - z: 0
- if layout2:
  - x: $tank_{origin0}$
  - y: $tank_{origin1}$ - $T_{PlantWall}$
  - z: $tank_{origin2}$ + H_{Sed} - $H_{EChannel}$
- if layout3:
  - x: $tank_{origin0}$ - L_{Sed} + $W_{Channel}$ + $T_{ChannelWall}$ + $W_{EChannel}$
  - y: $tank_{origin1}$ - $T_{PlantWall}$
  - z: $tank_{origin2}$ + H_{Sed} - $H_{EChannel}$

- if layout4:
  - x: $tank_{origin0}$ - L_{Sed} + $W_{ChannelInlet}$ + $T_{ChannelWall}$ + $W_{EChannel}$
  - y: $tank_{origin1}$ - $T_{PlantWall}$
  - z: $tank_{origin2}$ + H_{Sed} - $H_{EChannel}$

$echannelbox_{dim}$ =

- x: $W_{EChannel}$
- y: $N_{SedTanks(W}$Sed~ + $T_{PlantWall}$)
- z: $H_{EChannel}$

subtract2 - SubtractD subtracts one object from the other based on two points.

subtract <- subtractD(p1,$echannelbox_{origin}$)

Note: p1 is a dummy variable used only in the program help section to designate the matrix below.

p1 =

- x: $Plant_{Origin0}$ + $T_{PlantWall}$
- y: $Plant_{Origin1}$
- z: $Plant_{Origin2}$

$echannel_{origin}$ =

- if layout1:
  - x: 0
  - y: 0
  - z: 0
- if layout2:
  - x: $tank_{origin0}$
  - y: $tank_{origin1}$ - $T_{PlantWall}$
  - z: $tank_{origin2}$ + H_{Sed} - $H_{EChannel}$
- if layout3:
  - x: $tank_{origin0}$ - L_{Sed} + $W_{Channel}$ + $T_{ChannelWall}$ + $W_{EChannel}$
  - y: $tank_{origin1}$ - $T_{PlantWall}$
  - z: $tank_{origin2}$ + H_{Sed} - $H_{EChannel}$
- if layout4:
  - x: $tank_{origin0}$ - L_{Sed} + $W_{ChannelInlet}$ + $T_{ChannelWall}$ + $W_{EChannel}$
  - y: $tank_{origin1}$ - $T_{PlantWall}$
  - z: $tank_{origin2}$ + H_{Sed} - $H_{EChannel}$

layerset1 - $Layer_{set}$ selects the layer "channel."

layerset1 <- $layer_{set}$("channel")

box3 - Creates a box based on two points.

box3 <- box($channelclosebox_{origin}$,$channelclosebox_{origin}$ + $channelclosebox_{dim}$)

$channelclosebox_{origin}$ =

- x: $tank_{origin0}$ - L_{Sed} - $T_{PlantWall}$ - $W_{Channel}$ - $T_{ChannelWall}$
- y: $tank_{origin1}$ - $T_{PlantWall}$
- z: $tank_{origin2}$ + H_{Sed} - $H_{Channel}$ - $T_{ChannelWall}$

$channelclosebox_{dim}$ =

- x: $W_{Channel}$ + $T_{ChannelWall}$ + T{~}PlantWall
- y: T{~}PlantWall

- z: $H_{Channel} + T_{ChannelWall}$

bigunion - $Union_{allA}$ selects all the objects visible in the workspace and joins them into one single object.

bigunion <- $union_{allA}$

layerset - $Layer_{set}$ selects the layer "0".

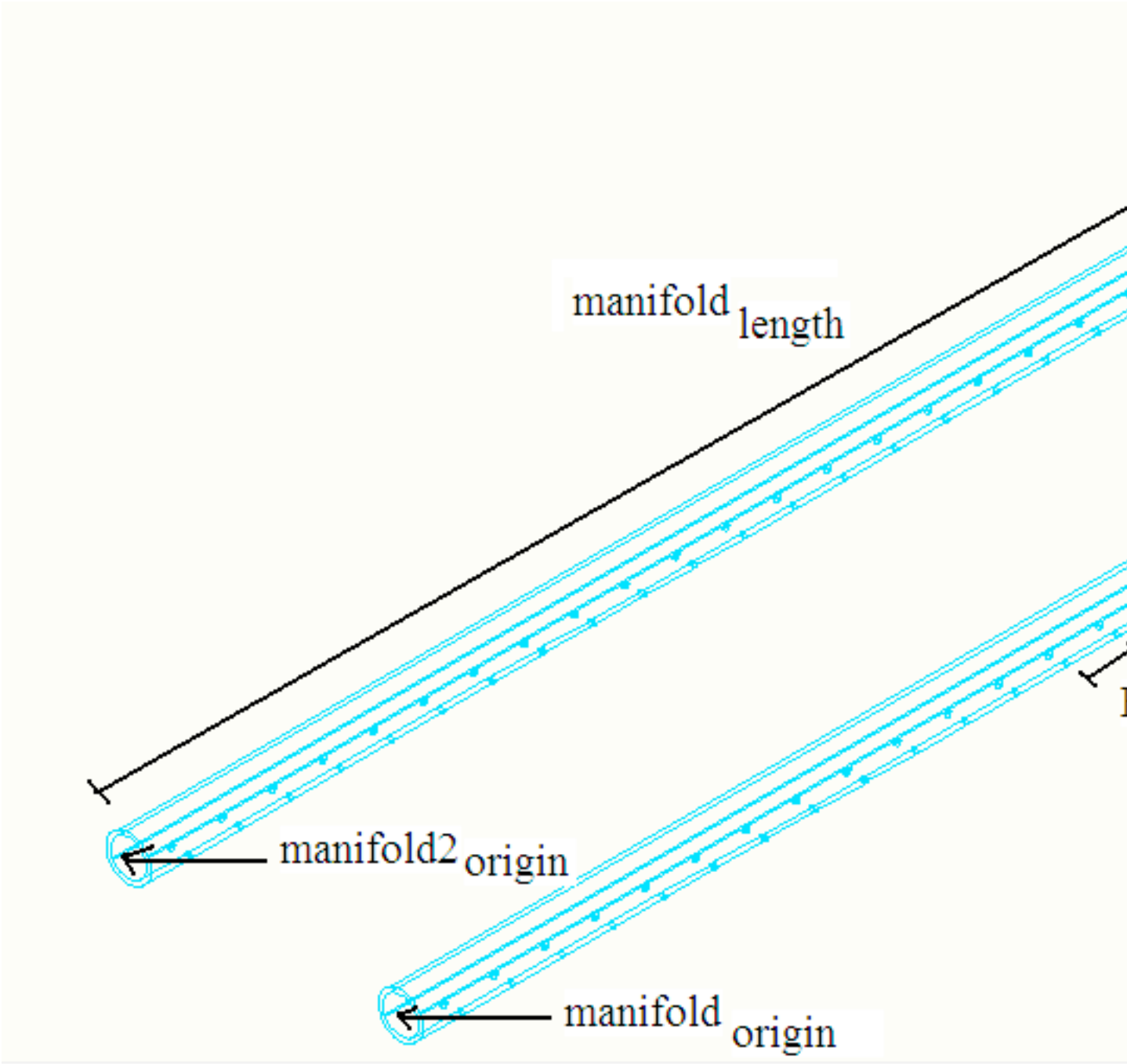
layerset <- $layer_{set}$("0")

layerfreezech - $Layer_{freeze}$ locks the layer "channel" so that it cannot be edited.

layerfreezech <- $layer_{freeze}$("channel")

layerfreezetank - $Layer_{freeze}$ locks the layer "slopes" so that it cannot be edited.

layerfreezetank <- $Layer_{freeze}$("slopes")

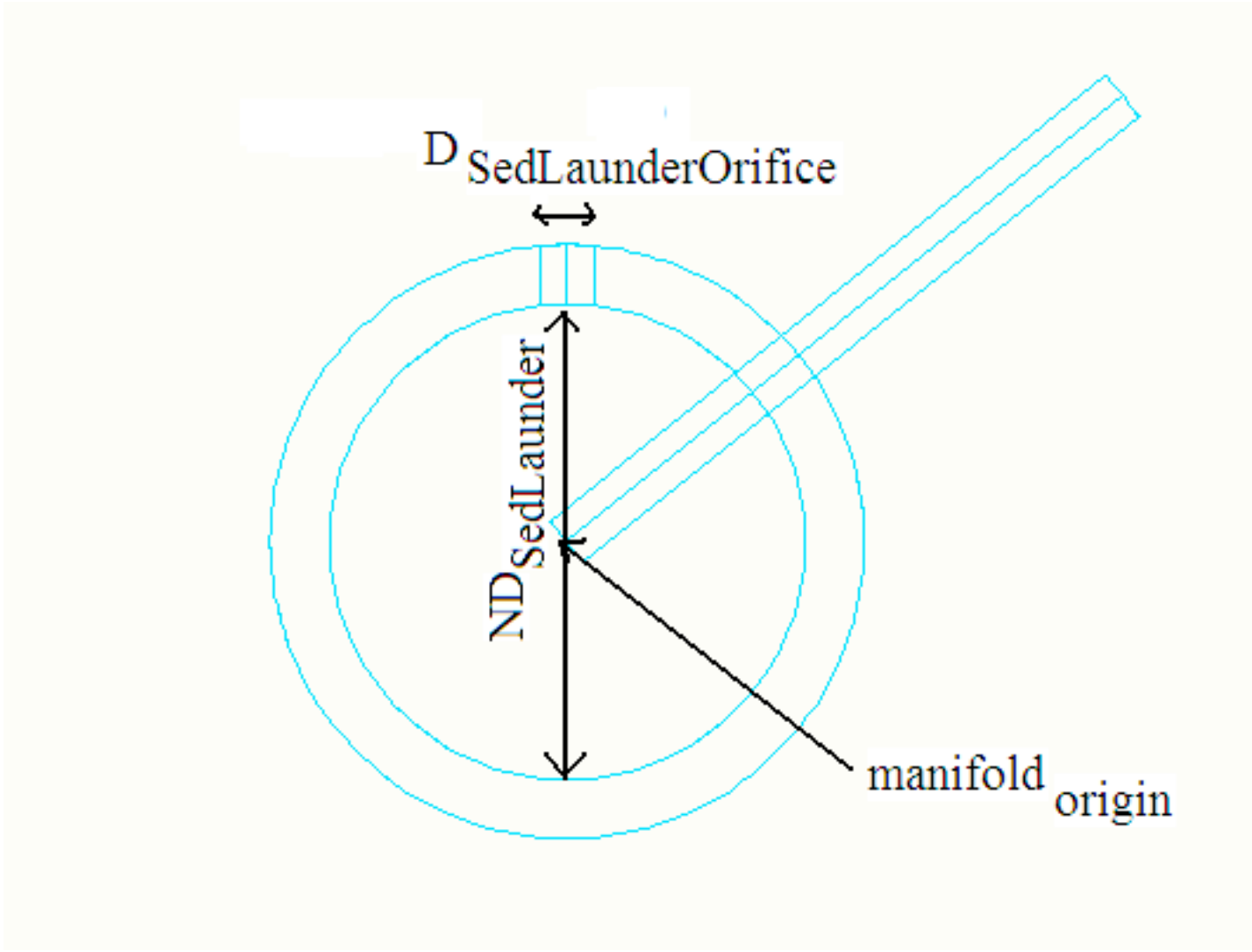


NorthEast Isometric View

Manifold Drawing Script

layer5 - Layer$_{new}$ creates a new blue layer "manifold."

layer5 <- layer$_{new}$("manifold',blue2)



Top View

manifold1 - Calls the Manifold Program to create a manifold.

manifold1 <- Manifold($manifold_{origin}$,$ND_{SedLaunder}$,$D_{SedLaunderOrifice}/2$,$manifold_{length}$,$B_{SedLaunderOrifices}$,$EN_{PipeSpec}$)
$manifold_{origin}$ =

- if layout 1:
  - x: $tank_{origin0}$ - L_{Sed} + $W_{Channel}$ + $T_{ChannelWall}$ + 2*ElbowRadius($ND_{SedLaunder}$)
  - y: $tank_{origin1}$ + $W_{Sed}/2$
  - z: $tank_{origin2}$ + HW_{Sed} - $H_{SedAbove}/2$
- if layout 2:
  - x: $tank_{origin0}$ - L_{Sed} + $W_{Channel}$ + $T_{ChannelWall}$
  - y: $tank_{origin1}$ + $W_{Sed}/3$
  - z: $tank_{origin2}$ + HW_{Sed} - $H_{SedAbove}/2$
- if layout 3:
  - x: $tank_{origin0}$ - L_{Sed} + $W_{Channel}$ + $T_{ChannelWall}$ + $W_{EChannel}$
  - y: $tank_{origin1}$ + $W_{Sed}/3$
  - z: $tank_{origin2}$ + HW_{Sed} - $H_{SedAbove}/2$
- if layout 4:
  - x: $tank_{origin0}$ - L_{Sed} + $W_{ChannelInlet}$ + $T_{ChannelWall}$ + $W_{EChannel}$
  - y: $tank_{origin1}$ + $W_{Sed}/3$

- z: $tank_{origin2}$ + HW_{Sed} - $H_{SedAbove}/2$

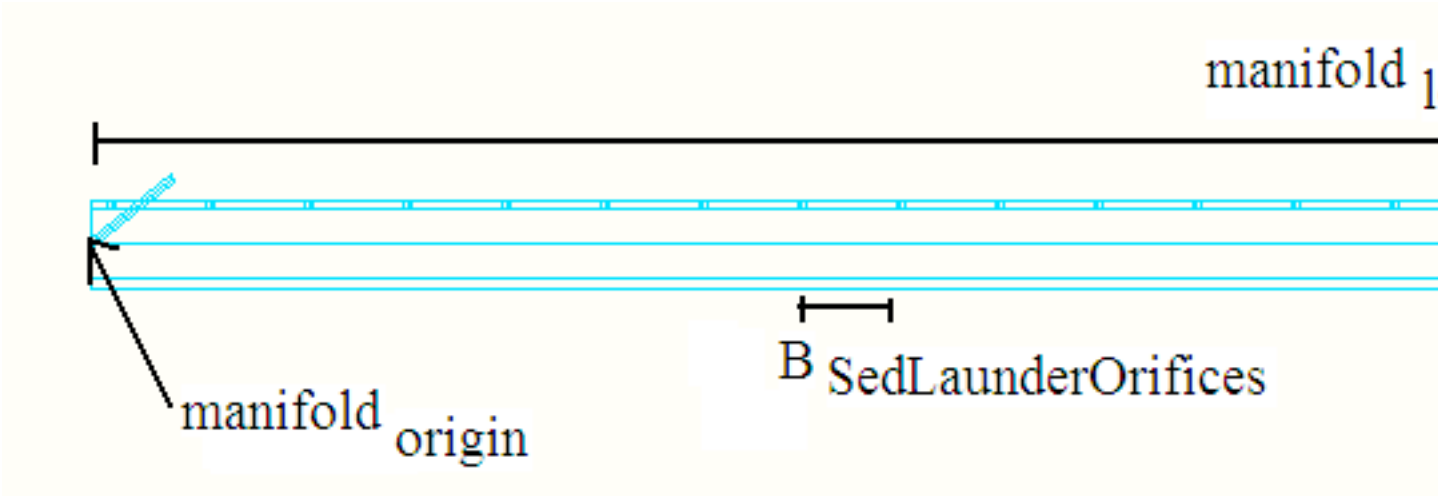
$ND_{SedLaunder}$ - Nominal diameter of launder.

$D_{SedLaunderOrifice}$ - Diameter of launder orifices.

$manifold_{length}$ - Specifies manifold length.

$B_{SedLaunderOrifices}$ - Spacing between orifices throughout the launder.

$EN_{PipeSpec}$ - Enumerated type.



Top View

rotate18 - [Rotate$_{3d}$] turns the object based on a given axis and degree angle.

rotate18 <- $rotate_{3d}$(p1,$manifold_{origin}$,"y",90)

p1 =

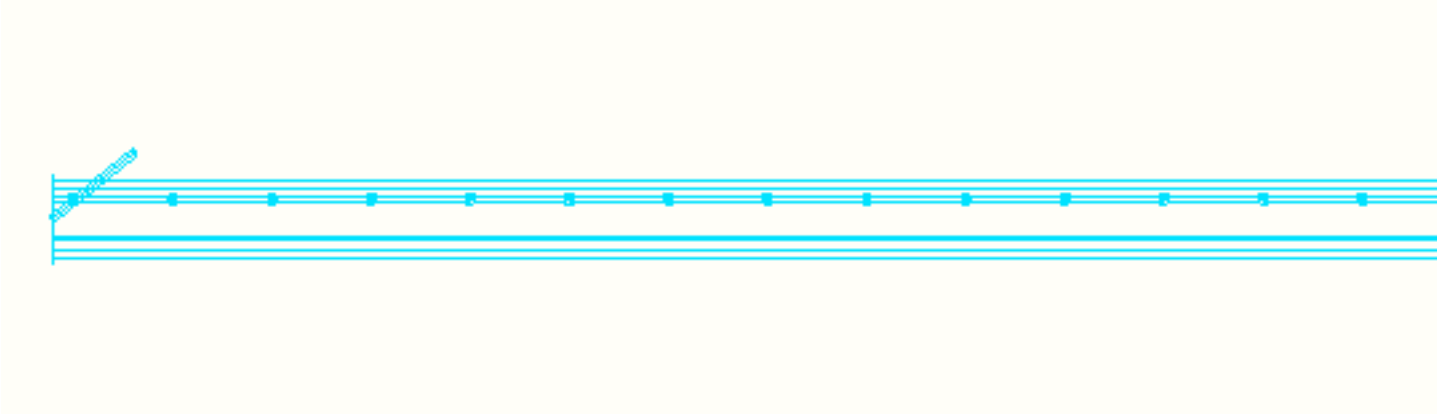
- x: $manifold_{origin0}$ + innerD($ND_{sedLaunder}$,$EN_{PipeSpec}$)/2
- y: $manifold_{origin1}$
- z: $manifold_{origin2}$

$manifold_{origin}$ =

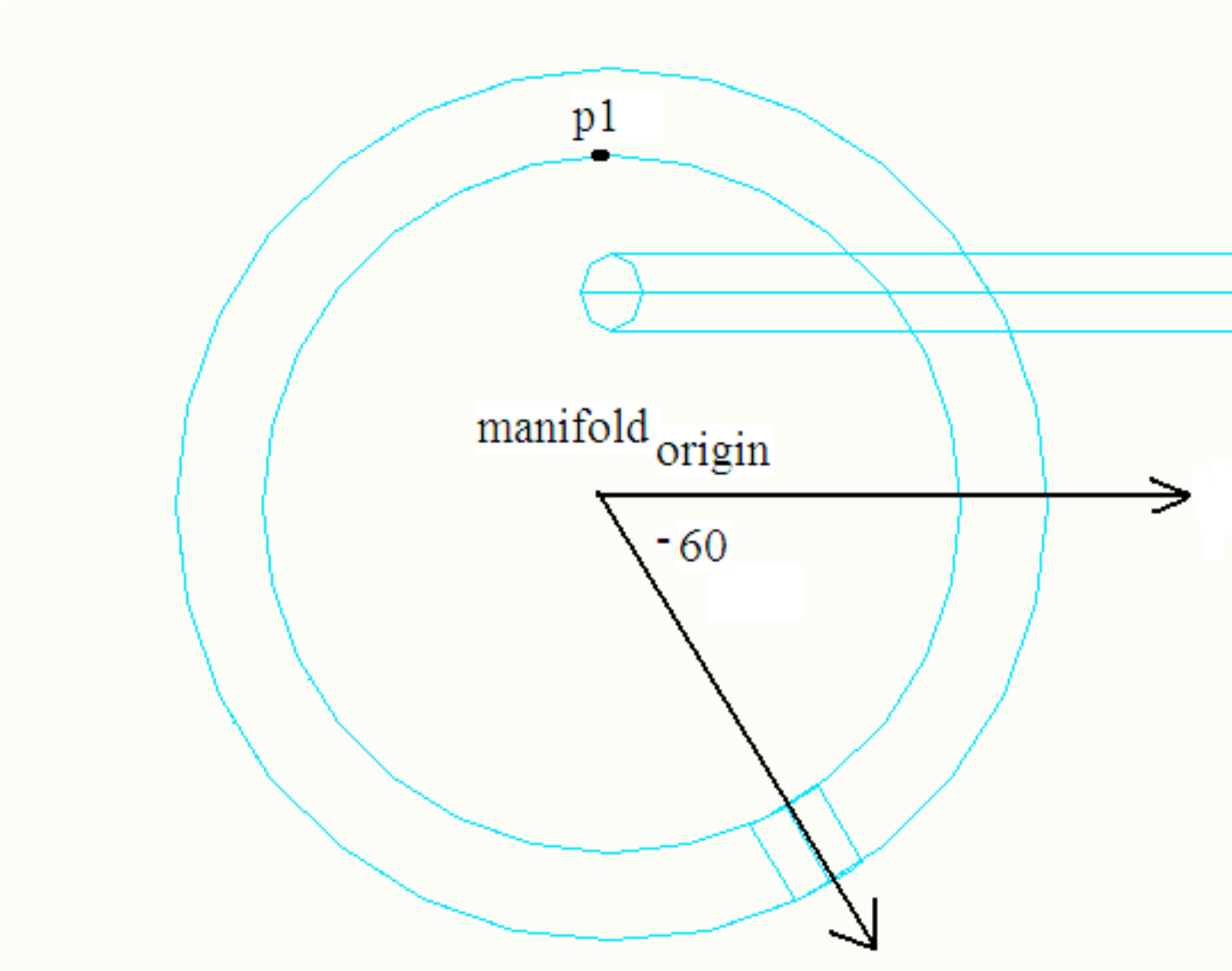
- if layout 1:
  - x: $tank_{origin0}$ - L_{Sed} + $W_{Channel}$ + $T_{ChannelWall}$ + 2*ElbowRadius($ND_{SedLaunder}$)
  - y: $tank_{origin1}$ + $W_{Sed}/2$
  - z: $tank_{origin2}$ + HW_{Sed} - $H_{SedAbove}/2$
- if layout 2:
  - x: $tank_{origin0}$ - L_{Sed} + $W_{Channel}$ + $T_{ChannelWall}$
  - y: $tank_{origin1}$ + $W_{Sed}/3$
  - z: $tank_{origin2}$ + HW_{Sed} - $H_{SedAbove}/2$
- if layout 3:
  - x: $tank_{origin0}$ - L_{Sed} + $W_{Channel}$ + $T_{ChannelWall}$ + $W_{EChannel}$
  - y: $tank_{origin1}$ + $W_{Sed}/3$
  - z: $tank_{origin2}$ + HW_{Sed} - $H_{SedAbove}/2$
- if layout 4:
  - x: $tank_{origin0}$ - L_{Sed} + $W_{ChannelInlet}$ + $T_{ChannelWall}$ + $W_{EChannel}$
  - y: $tank_{origin1}$ + $W_{Sed}/3$
  - z: $tank_{origin2}$ + HW_{Sed} - $H_{SedAbove}/2$

"y" - specifies axis that object will be rotated about.

90 - rotation angle



Top View



Right View

rotate19 - Rotate$_{3d}$ turns the object based on a given axis and degree angle.

rotate19 <- rotate$_{3d}$(p1,manifold$_{origin}$,"x",-60)

p1 =

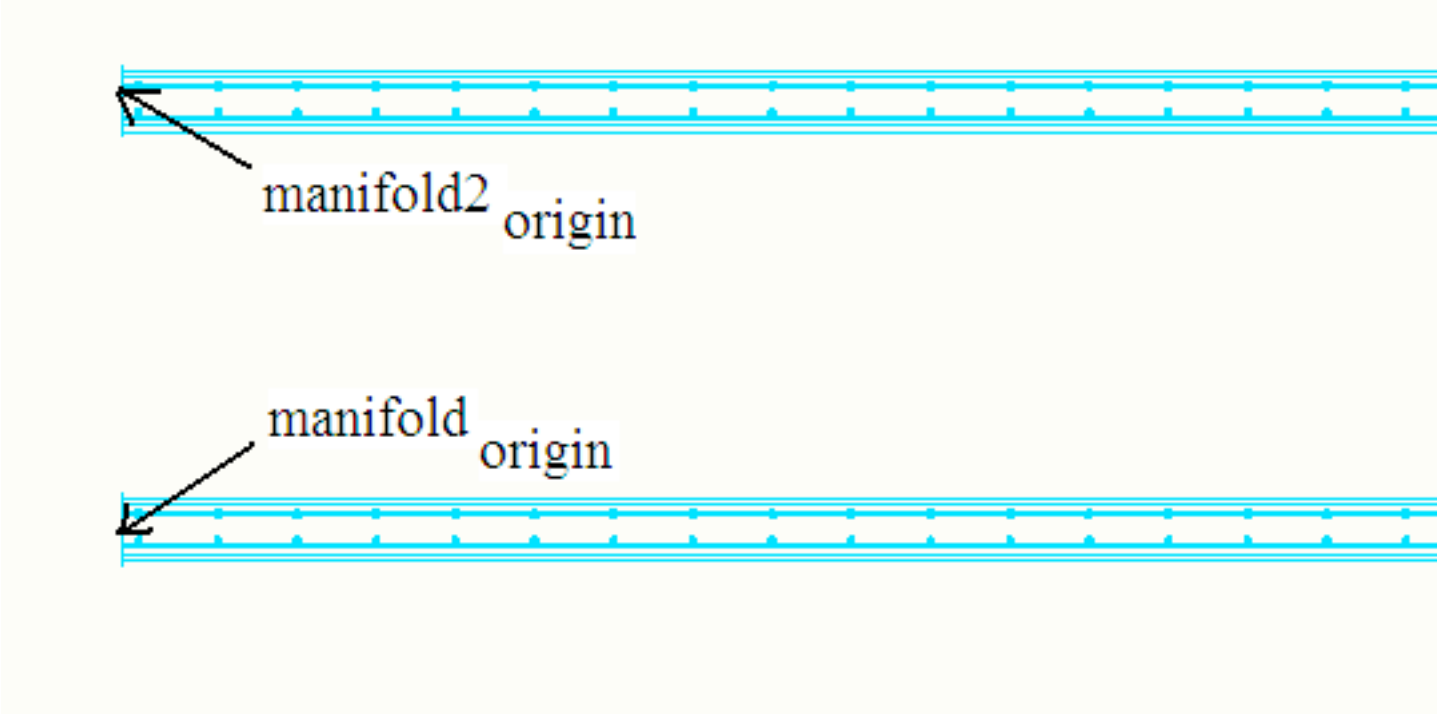
- x: $manifold_{origin0}$
- y: $manifold_{origin1}$ + innerD($ND_{sedLaunder}$,$EN_{PipeSpec}$)/2
- z: $manifold_{origin2}$

$manifold_{origin}$ =

- if layout 1:
  - x: $tank_{origin0}$ - L_{Sed} + $W_{Channel}$ + $T_{ChannelWall}$ + 2*ElbowRadius($ND_{SedLaunder}$)
  - y: $tank_{origin1}$ + W_{Sed}/2
  - z: $tank_{origin2}$ + HW_{Sed} - $H_{SedAbove}$/2
- if layout 2:
  - x: $tank_{origin0}$ - L_{Sed} + $W_{Channel}$ + $T_{ChannelWall}$
  - y: $tank_{origin1}$ + W_{Sed}/3
  - z: $tank_{origin2}$ + HW_{Sed} - $H_{SedAbove}$/2
- if layout 3:
  - x: $tank_{origin0}$ - L_{Sed} + $W_{Channel}$ + $T_{ChannelWall}$ + $W_{EChannel}$
  - y: $tank_{origin1}$ + W_{Sed}/3
  - z: $tank_{origin2}$ + HW_{Sed} - $H_{SedAbove}$/2
- if layout 4:
  - x: $tank_{origin0}$ - L_{Sed} + $W_{ChannelInlet}$ + $T_{ChannelWall}$ + $W_{EChannel}$
  - y: $tank_{origin1}$ + W_{Sed}/3
  - z: $tank_{origin2}$ + HW_{Sed} - $H_{SedAbove}$/2

"x" - specifies axis that object will be rotated about.

-60 - rotation angle



Top View

copym - CopyB duplicates the selected object.

copym <- copyB($manifold_{origin}$,$manifold_{origin}$,$manifold2_{origin}$)

$manifold_{origin}$ =

- if layout 1:

    - x: $tank_{origin0}$ - L_{Sed} + $W_{Channel}$ + $T_{ChannelWall}$ + 2*ElbowRadius($ND_{SedLaunder}$)
    - y: $tank_{origin1}$ + W_{Sed}/2
    - z: $tank_{origin2}$ + HW_{Sed} - $H_{SedAbove}$/2

- if layout 2:
    - x: $tank_{origin0}$ - L_{Sed} + $W_{Channel}$ + $T_{ChannelWall}$
    - y: $tank_{origin1}$ + W_{Sed}/3
    - z: $tank_{origin2}$ + HW_{Sed} - $H_{SedAbove}$/2

- if layout 3:
    - x: $tank_{origin0}$ - L_{Sed} + $W_{Channel}$ + $T_{ChannelWall}$ + $W_{EChannel}$
    - y: $tank_{origin1}$ + W_{Sed}/3
    - z: $tank_{origin2}$ + HW_{Sed} - $H_{SedAbove}$/2

- if layout 4:
    - x: $tank_{origin0}$ - L_{Sed} + $W_{ChannelInlet}$ + $T_{ChannelWall}$ + $W_{EChannel}$
    - y: $tank_{origin1}$ + W_{Sed}/3
    - z: $tank_{origin2}$ + HW_{Sed} - $H_{SedAbove}$/2

$manifold2_{origin}$ =

- if layout 1:
    - x: 0
    - y: 0
    - z: 0

- if layout 2:
    - x: $tank_{origin0}$ - L_{Sed} + $W_{Channel}$ + $T_{ChannelWall}$
    - y: $tank_{origin1}$ + 2*W_{Sed}/3
    - z: $tank_{origin2}$ + HW_{Sed} - $H_{SedAbove}$/2

- if layout 3:
    - x: $tank_{origin0}$ - L_{Sed} + $W_{Channel}$ + $T_{ChannelWall}$ + $W_{EChannel}$
    - y: $tank_{origin1}$ + 2*W_{Sed}/3
    - z: $tank_{origin2}$ + HW_{Sed} - $H_{SedAbove}$/2

- if layout 4:
    - x: $tank_{origin0}$ - L_{Sed} + $W_{ChannelInlet}$ + $T_{ChannelWall}$ + $W_{EChannel}$
    - y: $tank_{origin1}$ + 2*W_{Sed}/3
    - z: $tank_{origin2}$ + HW_{Sed} - $H_{SedAbove}$/2
      layerset - $Layer_{set}$ selects the layer "0".

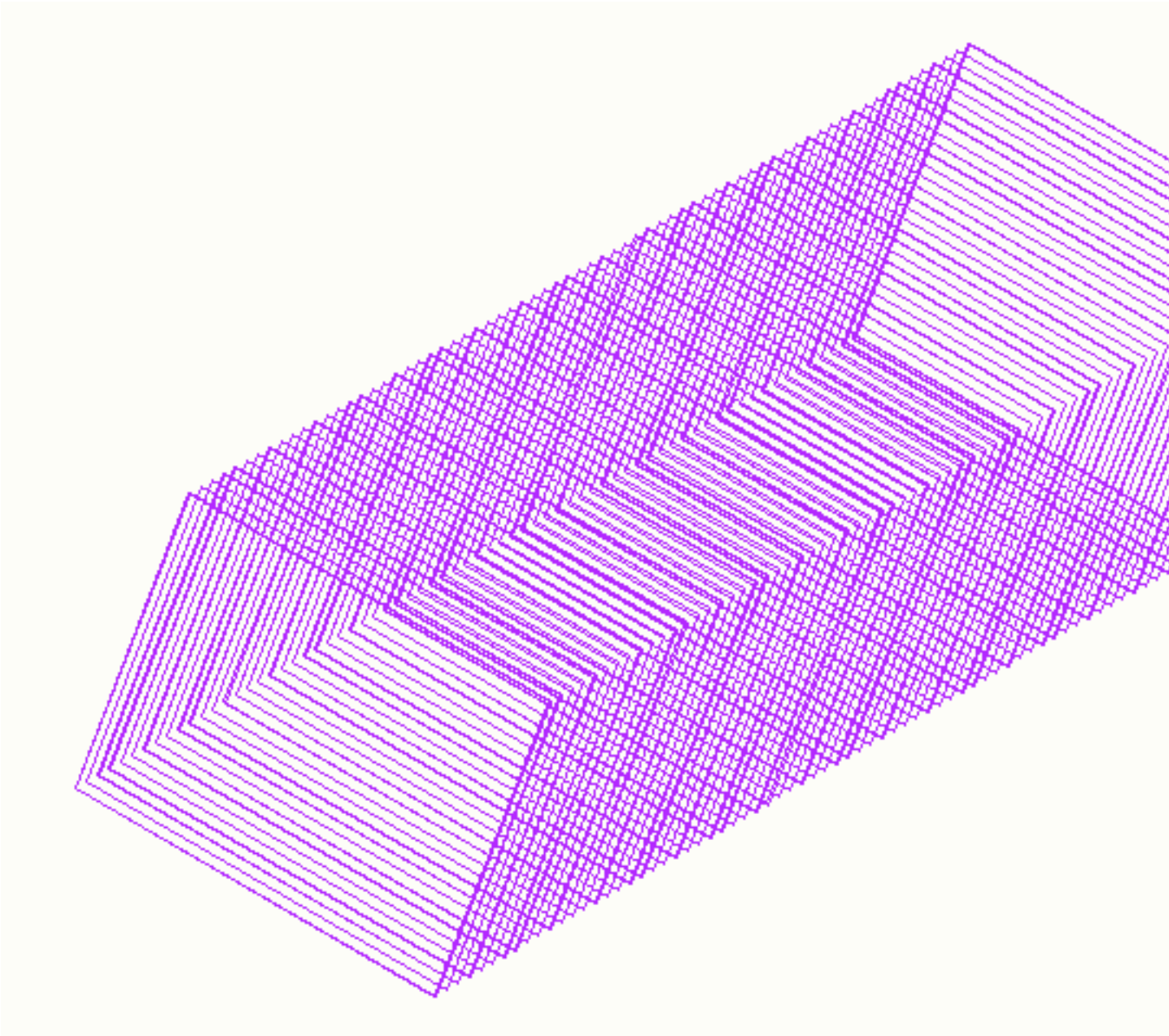
layerset <- $layer_{set}$("0")

layerfreeze5 - $Layer_{freeze}$ locks the layer "manifold" so that it cannot be edited.

layerfreeze5 <- $layer_{freeze}$("manifold")

AguaClara : AutoCAD Sedimentation Tank Program Sedtanklaminascript

This page last changed on Dec 18, 2008 by ar329.

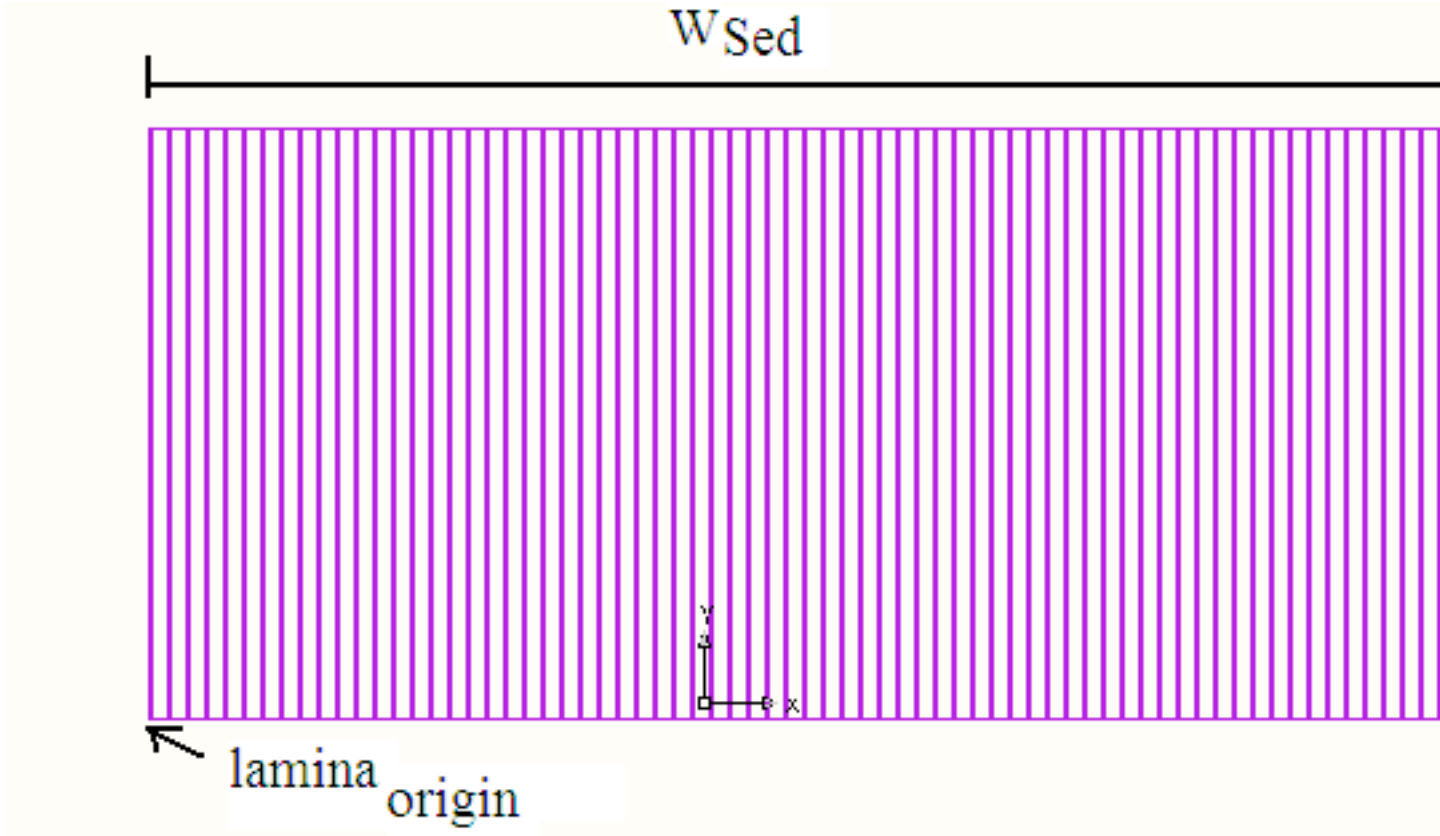


SouthWest Isometric View

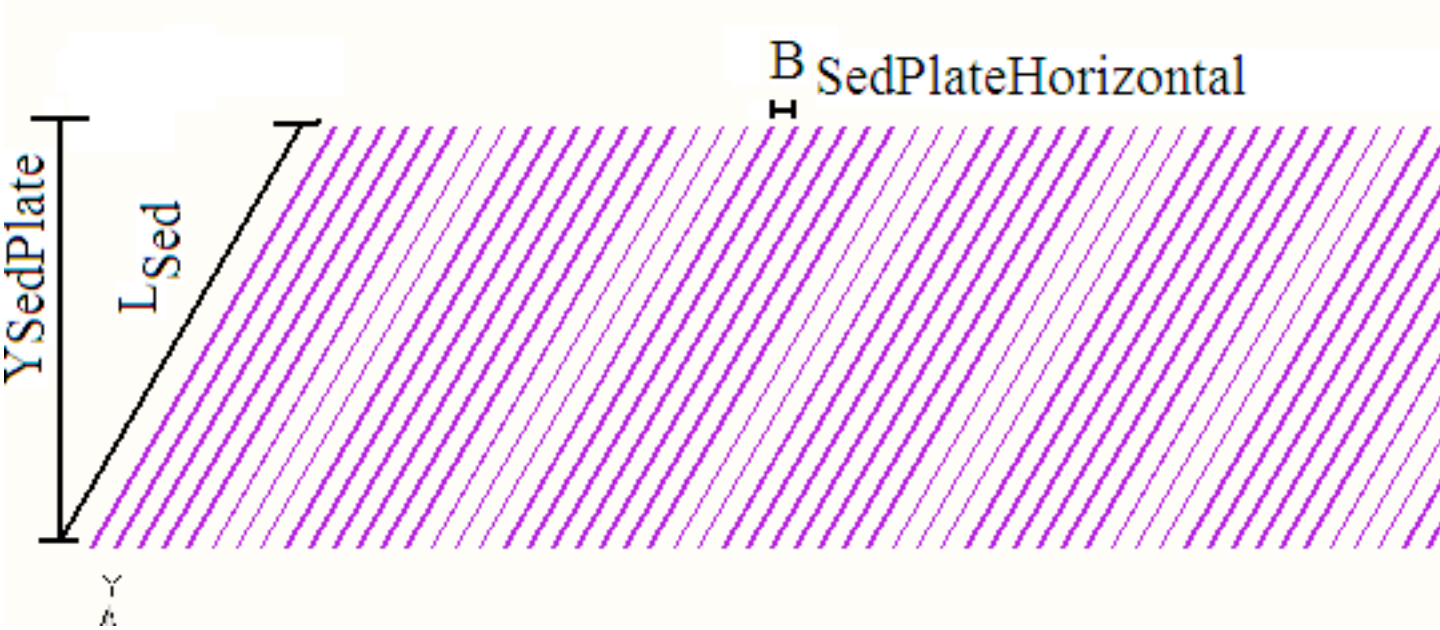
Lamina Drawing Script

layer7 - $\text{Layer}_{\text{new}}$ creates a new purple layer "lamina."

layer7 <- $\text{layer}_{\text{new}}$("lamina",purple)



Top View



Front View

lamina1 - Calls the Lamina Program to create lamella.

lamina1 <-
lamina($lamina_{origin}$,$Amp_{SedPlate}$,W_{Sed},$L_{SedPlate}$,$T_{SedPlate}$,$Per_{SedPlate}$,$AN_{SedPlate}$,$N_{SedPlates}$,$N_{SedPlateCol}$,-
$B_{SedPlateHorizontal}$,$Y_{SedPlate}$)

$lamina_{origin}$ =

- if layout1:

    - x: $tank_{origin0}$
    - y: $tank_{origin1}$
    - z: $tank_{origin2}$ + $H_{SedBelow}$
- if layout2:
    - x: $tank_{origin0}$ - $W_{EChannel}$
    - y: $tank_{origin1}$
    - z: $tank_{origin2}$ + $H_{SedBelow}$
- if layout3:
    - x: $tank_{origin0}$
    - y: $tank_{origin1}$
    - z: $tank_{origin2}$ + $H_{SedBelow}$
- if layout4:
    - x: $tank_{origin0}$
    - y: $tank_{origin1}$
    - z: $tank_{origin2}$ + $H_{SedBelow}$

$Amp_{SedPlate}$ - Degree lamella is installed relative to the horizontal (so flocs slide down lamella.)

W_{Sed} - Width of sedimentation tank.

$L_{SedPlate}$ - Length of plate settlers in the sedimentation tank.

$T_{SedPlate}$ - Thickness of plate settlers.

$Per_{SedPlate}$ - Period of the curves for the plate settlers.

$AN_{SedPlate}$ - Degree lamella is installed relative to the horizontal (so flocs slide down lamella.)

$N_{SedPlates}$ - Number of plate settlers in sed tank.

$N_{SedPlateCol}$ - Number of columns of sedimentation tank lamina when arrayed (always set to 1 since lamina are drawn one tank at a time.)

$B_{SedPlateHorizontal}$ - Horizontal spacing between sedimentation plates.

$Y_{SedPlate}$ - Vertical height of the sedimentation plates.
layerset - $Layer_{set}$ selects the layer "0".

layerset <- $layer_{set}$("0")

layerfreeze - $Layer_{freeze}$ locks the layer "lamina" so that it cannot be edited.

layerfreeze <- $layer_{freeze}$("lamina")

This page last changed on Nov 03, 2008 by ar329.

Inlet Slope Drawing Script

layerslopes2 - $Layer_{new}$ creates a new light grey layer "inletslopes."

layerslopes2 <- $layer_{new}$("inletslopes",ltgrey)

inletslopes - Calls the Sedimentation Tank Inlet Slopes Program to draw inlet slopes based on given inputs.

inletslopes <- Inletslope($inletslopes_{origin}$,L_{Sed},$L_{SedTopInlet}$,$L_{SedSlopeInlets}$,$H_{SedInlet}$,$T_{SedInletSlope}$,$N_{SedSlopeInlets}$,$AN_{SedTopInlet}$)

$inletslopes_{origin}$ =

- x: $Plant_{Origin0}$ - L_{Sed}
- y: $Plant_{Origin1}$ + W_{Sed}/2
- z: $Plant_{Origin2}$ + outerdiameter($ND_{SedSludge}$)

L_{Sed} = 2.658m

$L_{SedTopInlet}$ = 43.363deg

$L_{SedSlopeInlets}$ = $L_{SedInletSlope}$

$H_{SedInlet}$ = 0.057m

$T_{SedInletSlope}$ = 0.05m

$N_{SedSlopeInlets}$ = 1

$AN_{SedTopInlet}$ = 43.464deg

layerset - $Layer_{set}$ selects the layer "0".

layerset <- $layer_{set}$("0")

layerfreezeinslope - $Layer_{freeze}$ locks the layer "inletslopes" so that it cannot be edited.

layerfreezeinslope <- $layer_{freeze}$("inletslopes")

layerslopes - $Layer_{new}$ creates a new light grey layer "slopes."

layerslopes <- $layer_{new}$("slopes",ltgrey)

slopes - Calls the AutoCAD Sedimentation Tank Slopes Program to create tank slopes based on three inputs.

slopes <- sedslope($Plant_{Origin,p1,AN}$SedBottom~)

Note: p1 is a dummy variable used only in the program help section to designate the matrix below.

$Plant_{Origin}$ =

p1 =

- x: L_{Sed}
- y: W_{Sed}
- z: H_{Sed}

$AN_{SedBottom}$ = 10deg

moves - $Move_{all}$ shifts all the objects in the workspace.

moves <- $move_{all}(Plant_{Origin},p1)$

$Plant_{Origin}$ =

p1 =

- x: $Plant_{Origin0}$
- y: $Plant_{Origin1}$
- z: $Plant_{Origin2}$ + outerdiameter($ND_{SedSludge}$)

inletslopesridge - Calls the Sedimentation Tank Inlet Slopes Program to create inlet slopes based on given inputs.

inletslopesridge <- Inletslope($inletslopes_{origin}$,L_{Sed},$L_{sedTopInlet}$ + ($T_{SedInletSlope*tan(AN}$SedTopInlet~)),$L_{SedSlopeInlets}$,$L_{SedTopInlet}$,$T_{SedInletSlope}$,$N_{SedSlopeInlets}$,$AN_{SedTopInlet}$)

$inletslopes_{origin}$ =

- x: $Plant_{Origin0}$ - L_{Sed}
- y: $Plant_{Origin1}$ + W_{Sed}/2
- z: $Plant_{Origin2}$ + outerdiameter($ND_{SedSludge}$)

L_{Sed} = 2.658m

$L_{SedTopInlet}$ = 43.363deg

$T_{SedInletSlope}$ = 0.05m

tan($AN_{SedTopInlet}$) = tan(43.464deg)

$L_{SedSlopeInlets}$ = $L_{SedInletSlope}$

$N_{SedSlopeInlets}$ = 1

$AN_{SedTopInlet}$ = 43.464deg

tankthaw - $Layer_{thaw}$ unlocks the layer "tank" so that it can be edited.

tankthaw <- $layer_{thaw}$("tank")

bigunion - $Union_{allA}$ selects all the objects in the workspace and unions them into a single object.

bigunion <- $union_{allA}$

box1 - Creates a box based on two points.

box1 <- box($sedtankbox1_{origin}$,$sedtankbox1_{origin}$ + $sedtankbox1_{dim}$

$sedtankbox1_{origin}$ =

- x: $Plant_{Origin0}$ - L_{Sed}
- y: $Plant_{Origin1}$ + W_{Sed}/2 - outerradius($ND_{SedSludge}$)
- z: $Plant_{Origin2}$

$sedtankbox1_{dim}$ =

- x: L_{Sed}
- y: outerdiameter($ND_{SedSludge}$)
- z: 3outerdiameter($ND_{SedSludge}$)

subtractbox - SubtractK subtracts one object from the other based on three points.

subtractbox <- subtractK($Plant_{Origin}$ - zc,p1,p2)

p1 =

- x: $Plant_{Origin0}$ - $L_{Sed}/4$
- y: $Plant_{Origin1}$ + $W_{Sed}/4$
- z: $Plant_{Origin2}$ + $H_{Sed}/4$

p2 =

- x: $Plant_{Origin0}$
- y: $Plant_{Origin1}$ + $W_{Sed}/2$
- z: $Plant_{Origin2}$

layerfreezeslope - $Layer_{freeze}$ locks the layer "slopes" so that it cannot be edited.

layerfreezeslope <- $layer_{freeze}$("slopes")

This page last changed on Dec 12, 2008 by ar329.

Inlet Pipe Drawing Script - layout 1

viewtops - Adjusts the workspace so that the object is being viewed from the top.

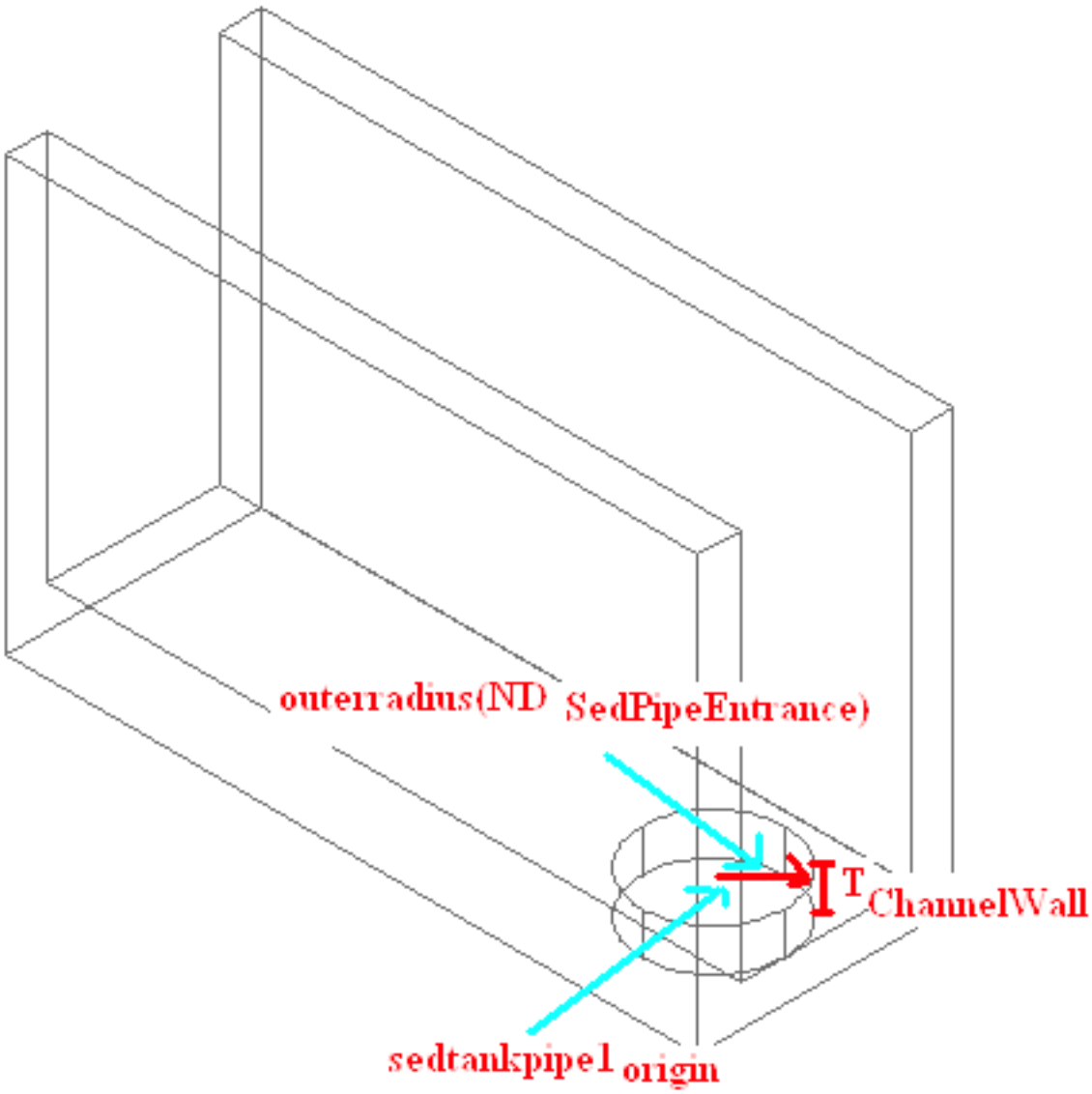
viewtops <- viewtop1

layerthawc - Layer$_{thaw}$ unlocks the layer "channel" so that it can be edited.

layerthawc <- layer$_{thaw}$("channel")

layersetc - Layer$_{set}$ selects the layer "channel".

layersetc <- layer$_{set}$("channel")



Northeast Isometric View

cylinder1 - CylinderA creates a cylinder based on three dimensions.

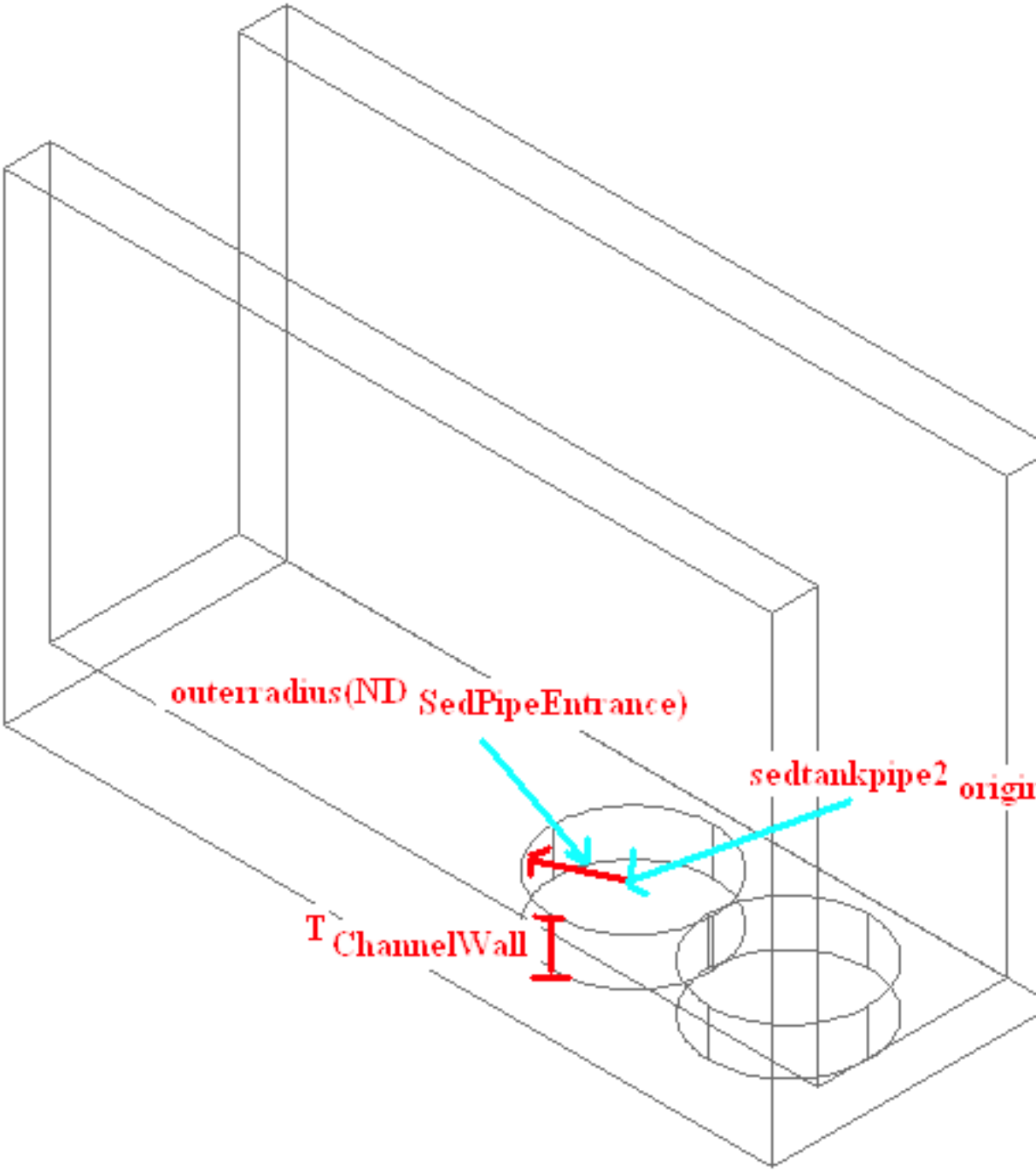
cylinder1 <- cylinderA(sedtankpipe1$_{origin}$,outerradius(ND$_{SedPipeEntrance}$),-T$_{ChannelWall}$)

sedtankpipe1$_{origin}$ =

- x: tank$_{origin0}$ - L$_{Sed}$ + W$_{Channel}$/2
- y: tank$_{origin1}$ + W$_{Sed}$/2 + outerradius(ND$_{SedLaunder}$) + 3*(W$_{Sed}$/2 - outerradius(ND$_{SedLaunder}$))/4
- z: tank$_{origin2}$ + H$_{Sed}$ - H$_{Channel}$

ND$_{SedPipeEntrance}$ = Nominal diameter of entrance pipe from the inlet channel to the sedimentation tank

T$_{ChannelWall}$ = Thickness of channel wall.



Northeast Isometric View

cylinder2 - CylinderA creates a cylinder based on three dimensions.

cylinder2 <- cylinderA(sedtankpipe2$_{origin}$,outerradius(ND$_{SedPipeEntrance}$),-T$_{ChannelWall}$)

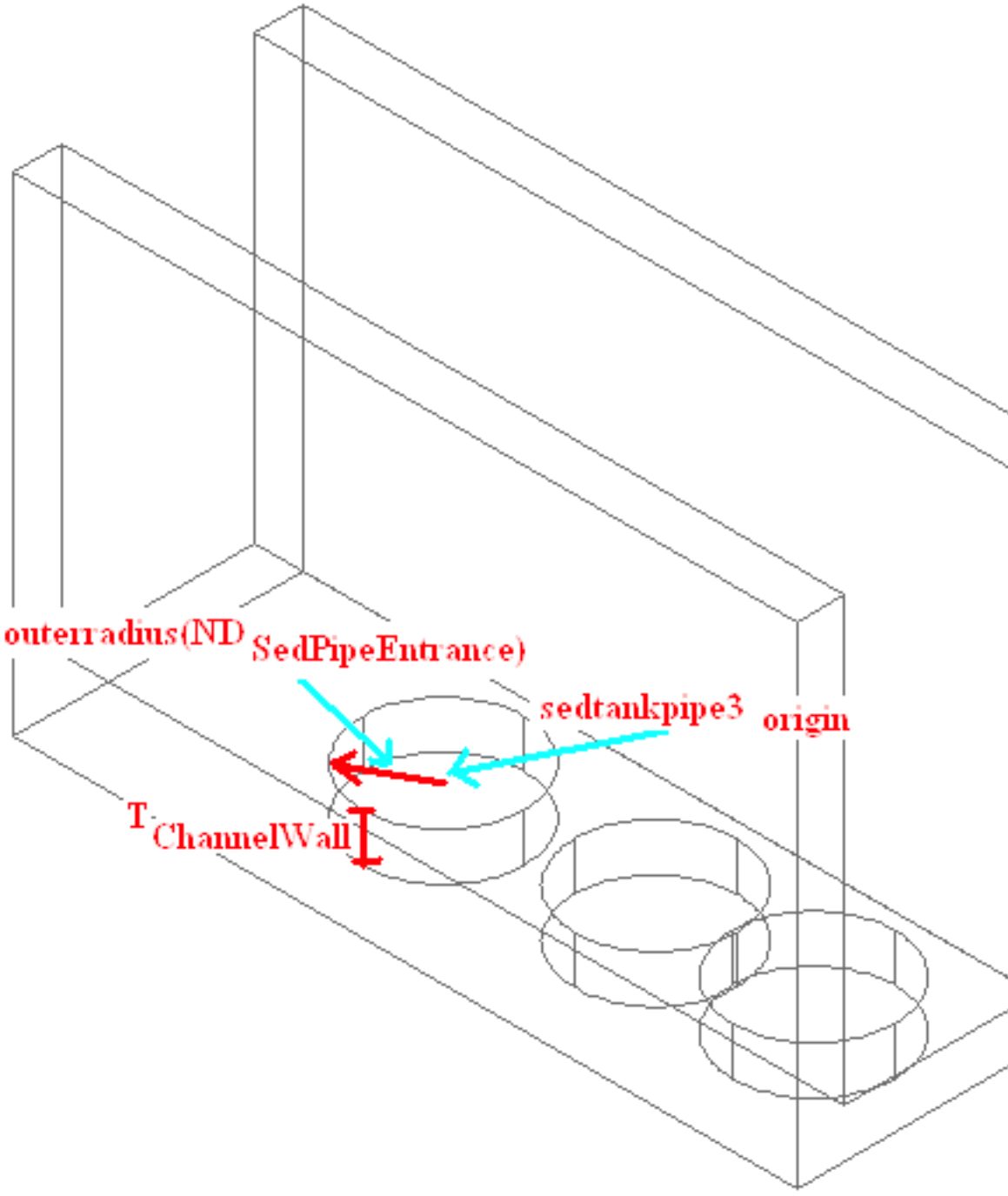
sedtankpipe2$_{origin}$ =

- x: tank$_{origin0}$ - L$_{Sed}$ + W$_{Channel}$/2
- y: tank$_{origin1}$ + W$_{Sed}$/2 + outerradius(ND$_{SedLaunder}$) + 1*(W$_{Sed}$/2 - outerradius(ND$_{SedLaunder}$))/4

- z: $tank_{origin2}$ + H_{Sed} - $H_{Channel}$

$ND_{SedPipeEntrance}$ = Nominal diameter of entrance pipe from the inlet channel to the sedimentation tank.

$T_{ChannelWall}$ = Thickness of channel wall.



Northeast Isometric View

cylinder3 - CylinderC creates a cylinder based on three dimensions.

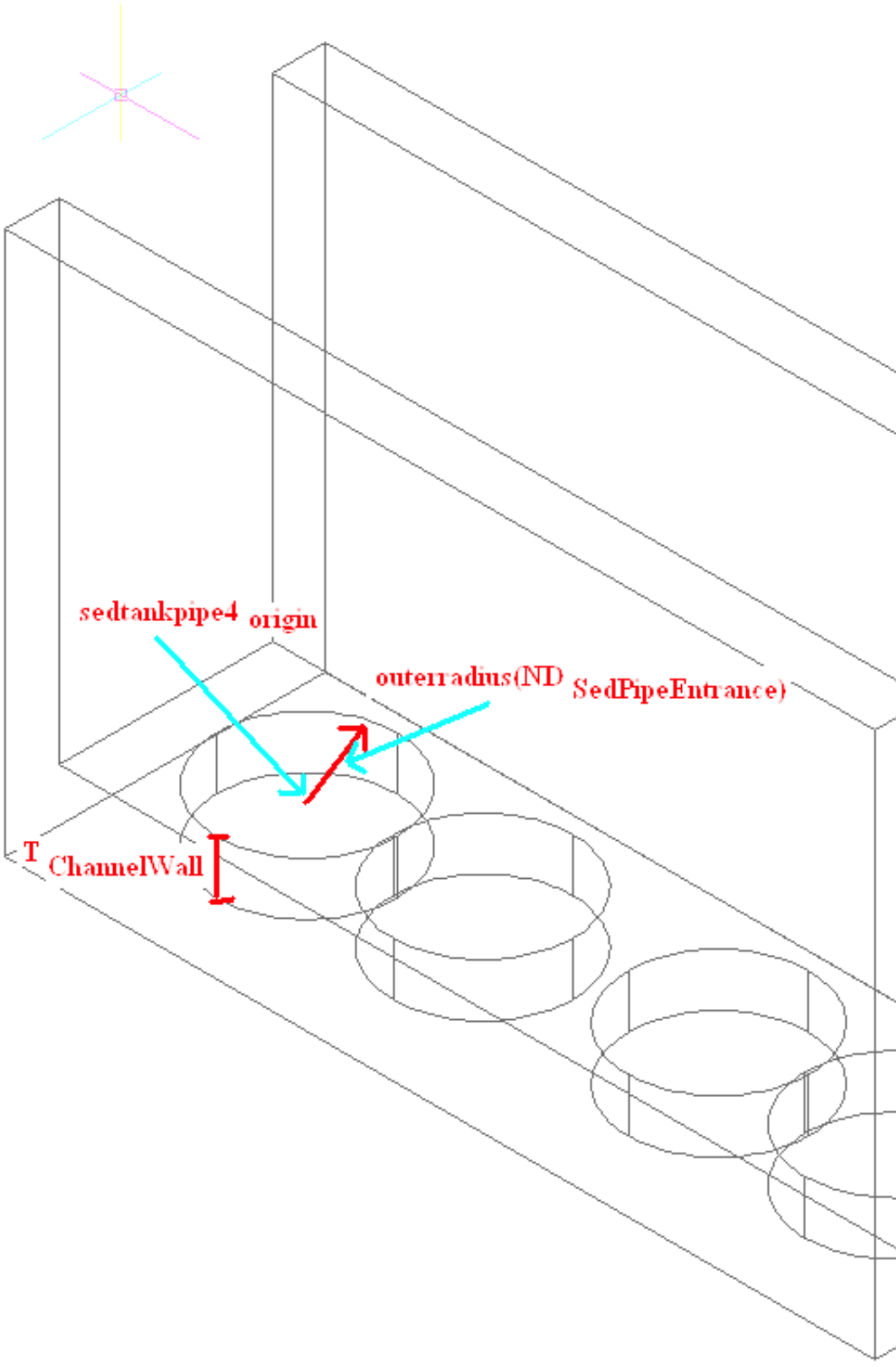
cylinder3 <- cylinderC(sedtankpipe3$_{origin}$,outerradius(ND$_{SedPipeEntrance}$),-T$_{ChannelWall}$)

sedtankpipe3$_{origin}$ =

- x: tank$_{origin0}$ - L$_{Sed}$ + W$_{Channel}$/2
- y: tank$_{origin1}$ + W$_{Sed}$/2 - outerradius(ND$_{SedLaunder}$) - 1*(W$_{Sed}$/2 - outerradius(ND$_{SedLaunder}$))/4
- z: tank$_{origin2}$ + H$_{Sed}$ - H$_{Channel}$

ND$_{SedPipeEntrance}$ = Nominal diameter of entrance pipe from the inlet channel to the sedimentation tank

T$_{ChannelWall}$ = Thickness of channel wall.


sedtankpipe4 origin
outerradius(ND SedPipeEntrance)
T ChannelWall

Northeast Isometric View

cylinder4 - CylinderA creates a cylinder based on three dimensions.

cylinder4 <- cylinderA(sedtankpipe4$_{origin}$,outerradius($ND_{SedPipeEntrance}$),-$T_{ChannelWall}$)

sedtankpipe4$_{origin}$ =

- x: $tank_{origin0}$ - L_{Sed} + $W_{Channel}$/2
- y: $tank_{origin1}$ + W_{Sed}/2 - outerradius($ND_{SedLaunder}$) - 3*(W_{Sed}/2 - outerradius($ND_{SedLaunder}$))/4
- z: $tank_{origin2}$ + H_{Sed} - $H_{Channel}$

$ND_{SedPipeEntrance}$ = Nominal diameter of entrance pipe from the inlet channel to the sedimentation tank

$T_{ChannelWall}$ = Thickness of channel wall.
subtract1 - SubtractH subtracts a selected object based on three specified points.

subtract1 <- subtractH(channel$_{origin}$,p1,p2)

channel$_{origin}$ =

- if layout 1-3:
  - x: $tank_{origin0}$ - L_{Sed} + $W_{Channel}$
  - y: $tank_{origin1}$ - $T_{PlantWall}$
  - z: $tank_{origin2}$ + H_{Sed} - $H_{Channel}$
- if layout4:
  - x: $tank_{origin0}$ - L_{Sed} -$T_{ChannelWall}$
  - y: $tank_{origin1}$ - $T_{PlantWall}$
  - z: $tank_{origin2}$ + H_{Sed} - $H_{Channel}$

p1 =

- x: sedtankpipe1$_{origin0}$ + outerradius($ND_{SedPipeEntrance}$)
- y: sedtankpipe1$_{origin1}$ + outerradius($ND_{SedPipeEntrance}$)
- z: sedtankpipe1$_{origin2}$

p2 =

- x: sedtankpipe4$_{origin0}$ - outerradius($ND_{SedPipeEntrance}$)
- y: sedtankpipe4$_{origin1}$ - outerradius($ND_{SedPipeEntrance}$)
- z: sedtankpipe4$_{origin2}$

layerset - Layer$_{set}$ selects the layer "0".

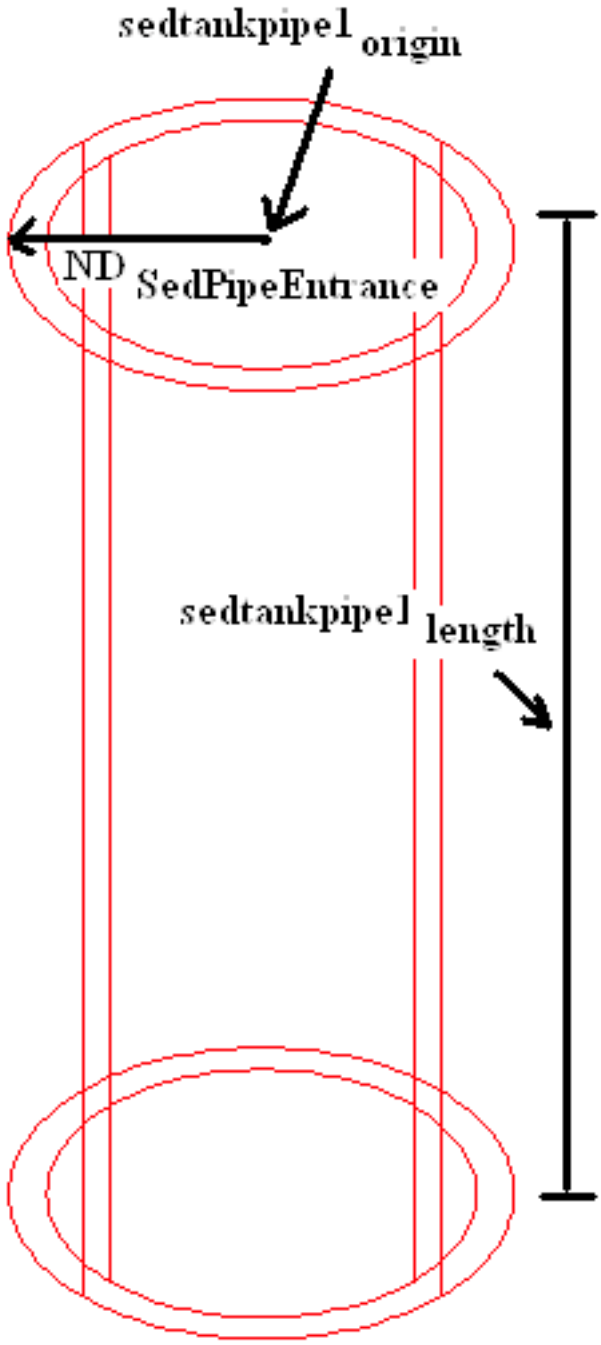
layerset <- layer$_{set}$("0")

layerfreeze - Layer$_{freeze}$ locks the layer "channel" so that it cannot be edited.

layerfreeze <- layer$_{freeze}$("channel")

layer3 - Layer$_{new}$ creates a new blue layer "pipe".

layer3 <- layer$_{new}$("pipe",blue)



Northeast Isometric View

pipe1 - Calls the Pipe Program to create a pipe.

pipe1 <- Pipe($sedtankpipe1_{origin}$,$ND_{SedPipeEntrance}$,-$sedtankpipe1_{length}$,$EN_{PipeSpec}$)

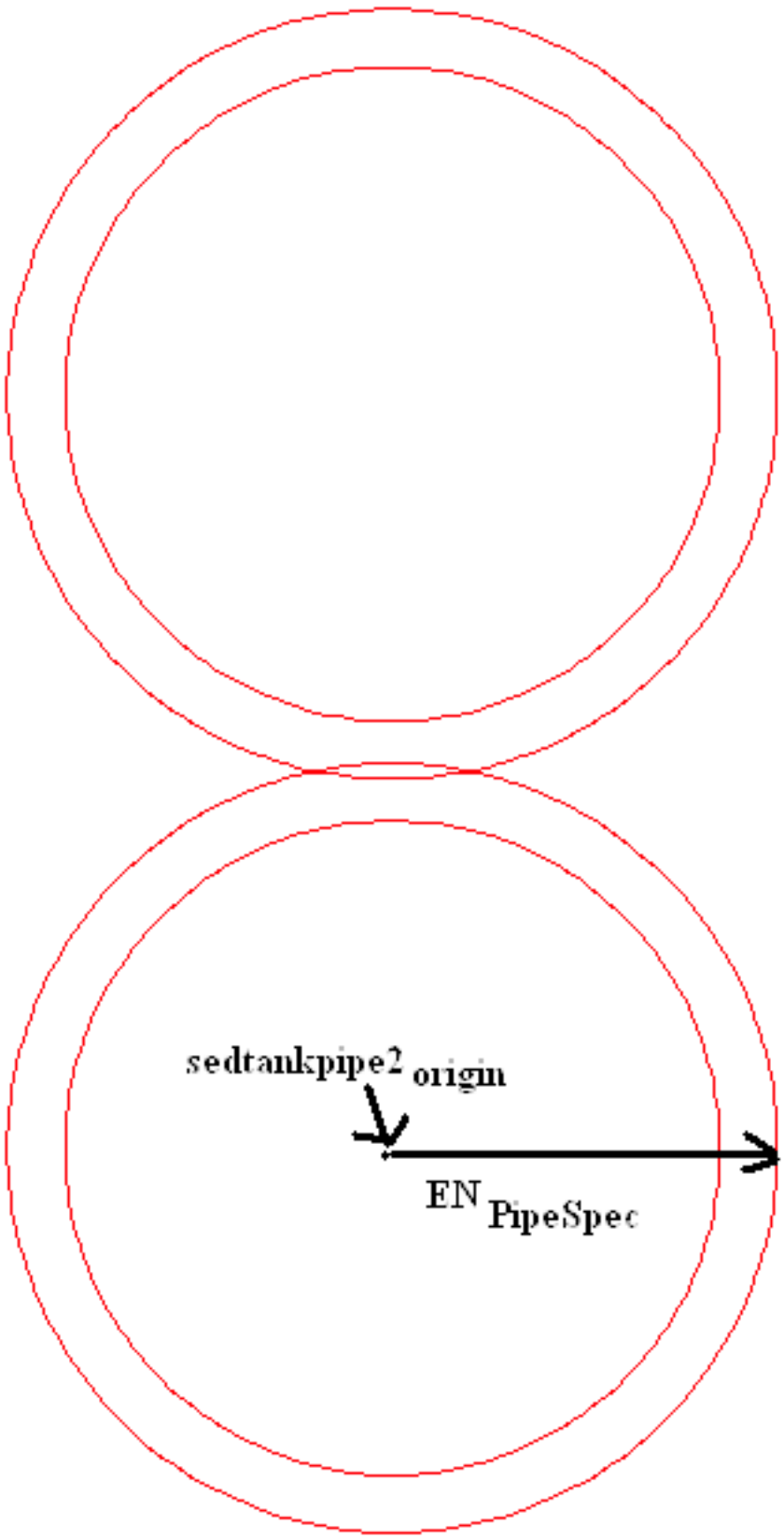
$sedtankpipe1_{origin}$ =

- x: $tank_{origin0}$ - L_{Sed} + $W_{Channel}/2$
- y: $tank_{origin1}$ + $W_{Sed}/2$ + outerradius($ND_{SedLaunder}$) + 3*($W_{Sed}/2$ - outerradius($ND_{SedLaunder}$))/4
- z: $tank_{origin2}$ + H_{Sed} - $H_{Channel}$

$ND_{SedPipeEntrance}$ = Nominal diameter of entrance pipe from the inlet channel to the sedimentation tank.

$sedtankpipe1\{\}_{length} = H_{Sed} - H_{InletChannel} - (W_{Sed}/2 * tank(AN_{SedBottom}) + ElbowRadius(ND_{SedPipeEntrance})) - ElbowRadius(ND_{SedPipeEntrance})$

$EN_{PipeSpec}$ = enumerated pipe schedule type. Each schedule of pipe is represented by a specific number within our code.


sedtankpipe2 origin
EN PipeSpec

Northeast Isometric View

pipe2 - Calls the Pipe Program to create a pipe.

pipe2 <- Pipe($sedtankpipe2_{origin}$,$ND_{SedPipeEntrance}$,-$sedtankpipe1_{length}$,$EN_{PipeSpec}$)

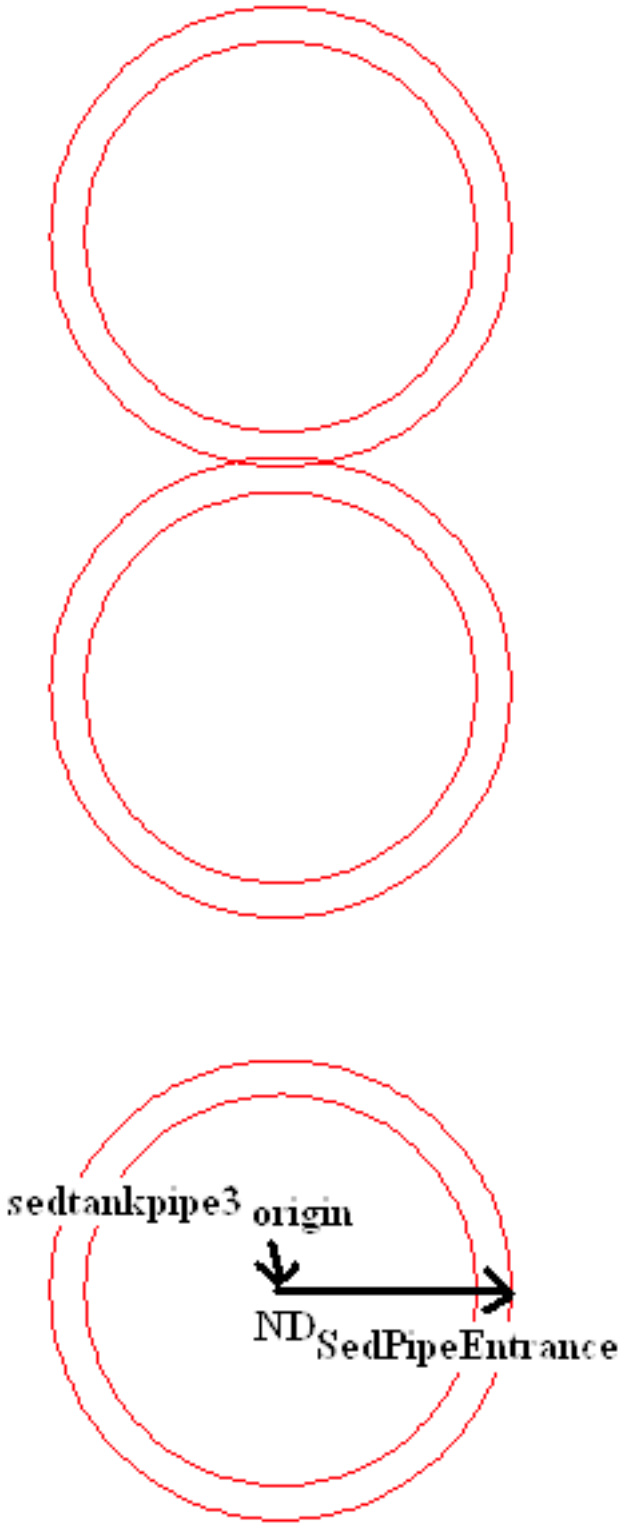
$sedtankpipe2_{origin}$ =

- x: $tank_{origin0}$ - L_{Sed} + $W_{Channel}/2$
- y: $tank_{origin1}$ + $W_{Sed}/2$ + outerradius($ND_{SedLaunder}$) + 1*($W_{Sed}/2$ - outerradius($ND_{SedLaunder}$))/4
- z: $tank_{origin2}$ + H_{Sed} - $H_{Channel}$

$ND_{SedPipeEntrance}$ = Nominal diameter of entrance pipe from the inlet channel to the sedimentation tank.

$sedtankpipe1\{\}_{length}$ = H_{Sed} - $H_{InletChannel}$ - ($W_{Sed}/2$ * tank($AN_{SedBottom}$) + ElbowRadius($ND_{SedPipeEntrance}$))-ElbowRadius($ND_{SedPipeEntrance}$)

$EN_{PipeSpec}$ = enumerated pipe schedule type. Each schedule of pipe is represented by a specific number within our code.



Northeast Isometric View

pipe3 - Calls the Pipe Program to create a pipe.

pipe3 <- Pipe($sedtankpipe3_{origin}$,$ND_{SedPipeEntrance}$,-$sedtankpipe1_{length}$,$EN_{PipeSpec}$)

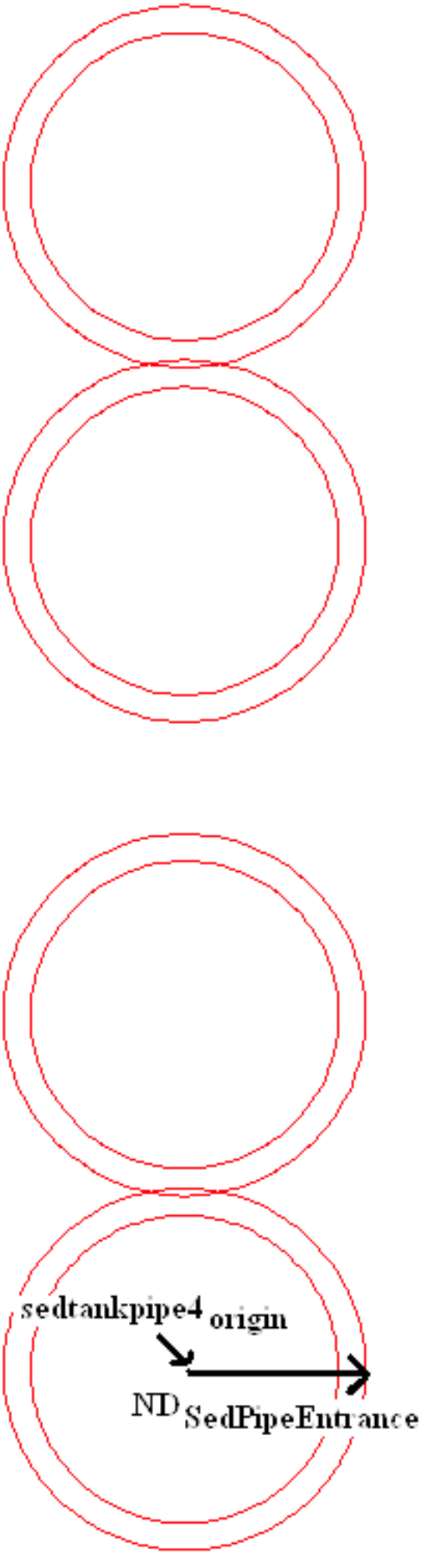
$sedtankpipe3_{origin}$ =

- x: $tank_{origin0}$ - L_{Sed} + $W_{Channel}/2$
- y: $tank_{origin1}$ + $W_{Sed}/2$ - outerradius($ND_{SedLaunder}$) - 1*($W_{Sed}/2$ - outerradius($ND_{SedLaunder}$))/4
- z: $tank_{origin2}$ + H_{Sed} - $H_{Channel}$

$ND_{SedPipeEntrance}$ = Nominal diameter of entrance pipe from the inlet channel to the sedimentation tank.

$sedtankpipe1\{\}_{length}$ = H_{Sed} - $H_{InletChannel}$ - ($W_{Sed}/2$ * tank($AN_{SedBottom}$) + ElbowRadius($ND_{SedPipeEntrance}$))-ElbowRadius($ND_{SedPipeEntrance}$)

$EN_{PipeSpec}$ = enumerated pipe schedule type. Each schedule of pipe is represented by a specific number within our code.


sedtankpipe4 origin
ND SedPipeEntrance

Northeast Isometric View

pipe4 - Calls the Pipe Program to create a pipe.

pipe4 <- Pipe($sedtankpipe4_{origin}$,$ND_{SedPipeEntrance}$,-$sedtankpipe1_{length}$,$EN_{PipeSpec}$)

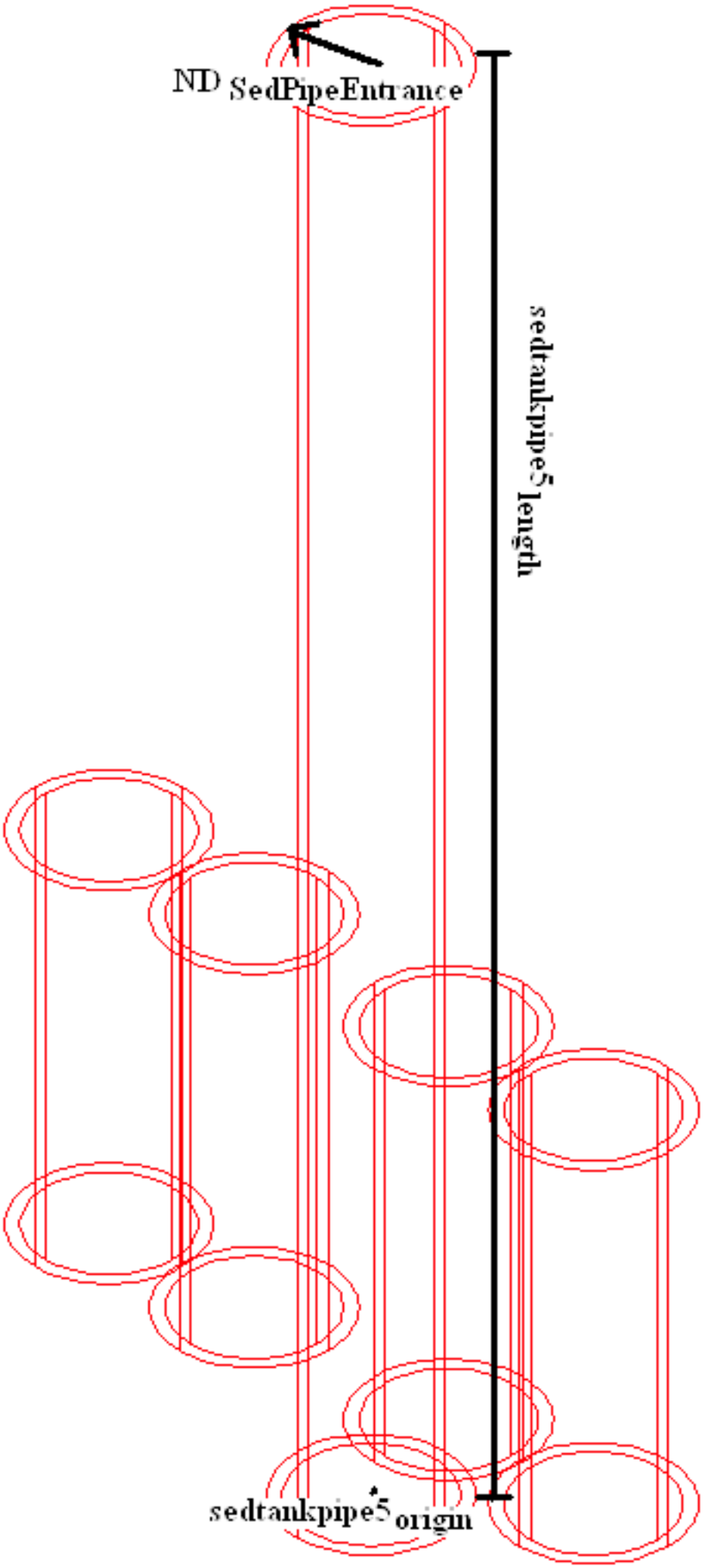
$sedtankpipe4_{origin}$ =

- x: $tank_{origin0}$ - L_{Sed} + $W_{Channel}/2$
- y: $tank_{origin1}$ + $W_{Sed}/2$ - outerradius($ND_{SedLaunder}$) - 3*($W_{Sed}/2$ - outerradius($ND_{SedLaunder}$))/4
- z: $tank_{origin2}$ + H_{Sed} - $H_{Channel}$

$ND_{SedPipeEntrance}$ = Nominal diameter of entrance pipe from the inlet channel to the sedimentation tank.

$sedtankpipe1\{\}_{length}$ = H_{Sed} - $H_{InletChannel}$ - ($W_{Sed}/2$ * tank($AN_{SedBottom}$) + ElbowRadius($ND_{SedPipeEntrance}$))-ElbowRadius($ND_{SedPipeEntrance}$)

$EN_{PipeSpec}$ = enumerated pipe schedule type. Each schedule of pipe is represented by a specific number within our code.


ND SedPipeEntrance
sedtankpipe5 length
sedtankpipe5 origin

Northeast Isometric View

pipe5 - Calls the Pipe Program to create a pipe.

pipe5 <- Pipe($sedtankpipe5_{origin}$,$ND_{SedPipeEntrance}$,-$sedtankpipe5_{length}$,$EN_{PipeSpec}$)

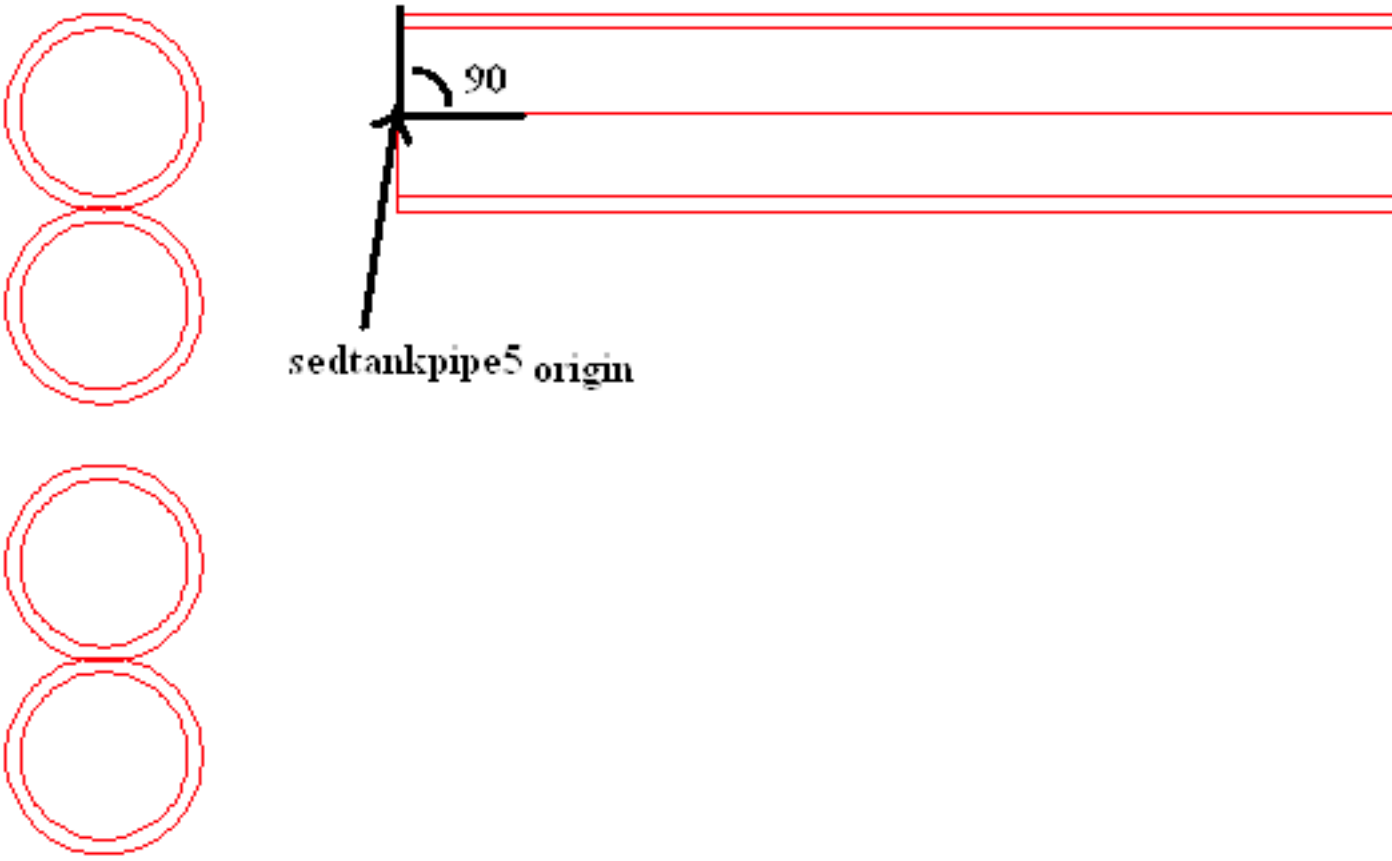
$sedtankpipe5_{origin}$ =

- x: $tank_{origin0}$ - L_{Sed} + $W_{Channel}/2$ + ElbowRadius($ND_{SedPipeEntrance}$)
- y: $tank_{origin1}$ + $W_{Sed}/2$ + outerradius($ND_{SedLaunder}$) + 3*($W_{Sed}/2$ - outerradius($ND_{SedLaunder}$))/4
- z: $tank_{origin2}$ + $W_{Sed}/2$*tan($AN_{SedBottom}$) + ElbowRadius($ND_{SedPipeEntrance}$)

$ND_{SedPipeEntrance}$ = Nominal diameter of entrance pipe from the inlet channel to the sedimentation tank.

$sedtankpipe5_{length}$ = $((5L_{Sed}/8 - W_{InletChannel}/2)^2$ + (outerradius($ND_{SedLaunder}$) + 3*($W_{Sed}/2$ - outerradius($ND_{SedLaunder}$))/4)$^2)^{1/2}$ - 2ElbowRadius(ND~SedPipeEntrance~)

$EN_{PipeSpec}$ = enumerated pipe schedule type. Each schedule of pipe is represented by a specific number within our code.



Northeast Isometric View

rotatepipe5 - Rotate$_{3d}$ turns the object based on a given axis and degree angle.

rotatepipe5 <- rotate$_{3d}$(p1,sedtankpipe5$_{origin}$,"y",90)

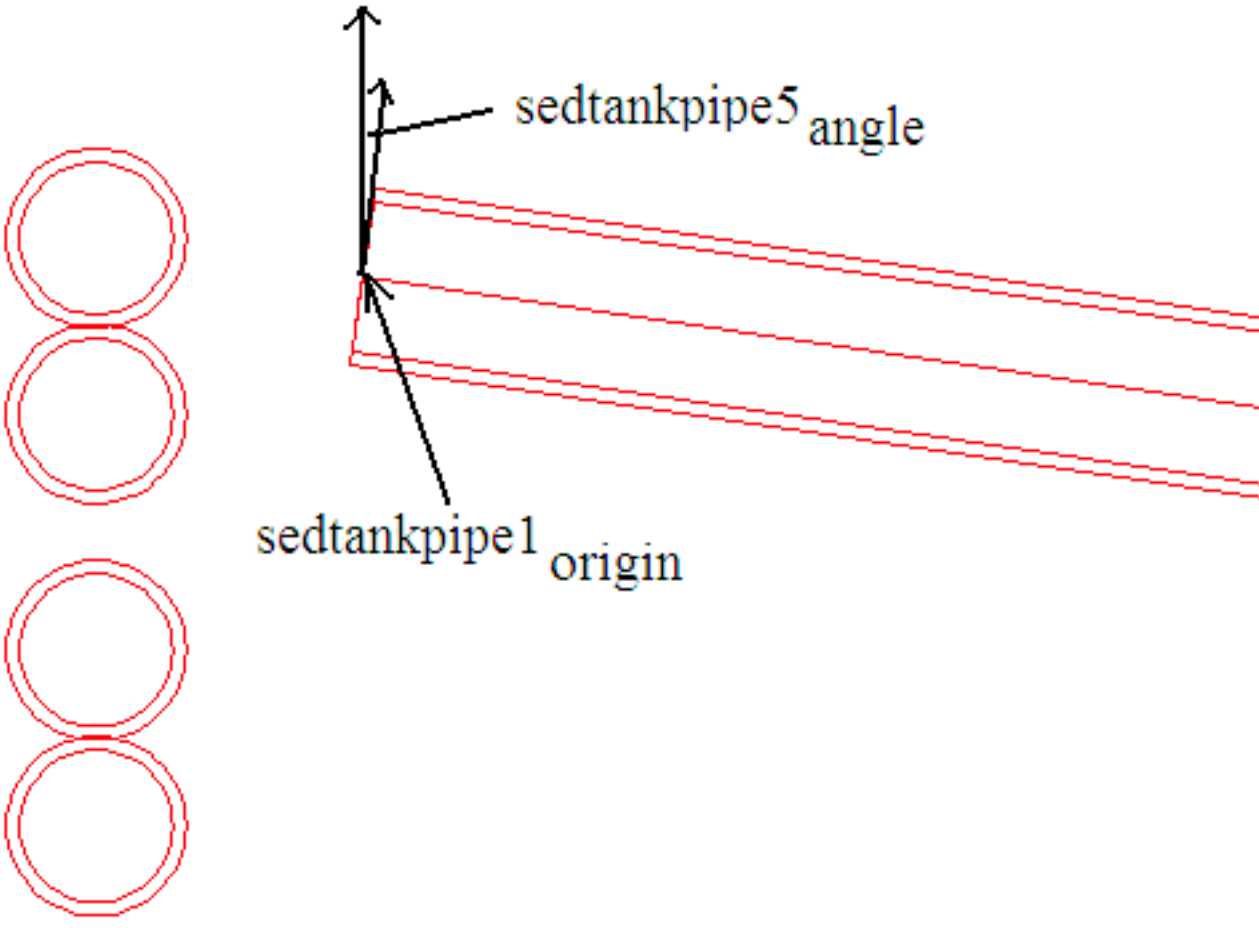
p1 =

- x: sedtankpipe5$_{origin0}$ + innerD(ND$_{SedPipeEntrance}$,EN$_{PipeSpec}$)/2
- y: sedtankpipe5$_{origin1}$
- z: sedtankpipe5$_{origin2}$

sedtankpipe5$_{origin}$ =

- x: tank$_{origin0}$ - L$_{Sed}$ + W$_{Channel}$/2 + ElbowRadius(ND$_{SedPipeEntrance}$)
- y: tank$_{origin1}$ + W$_{Sed}$/2 + outerradius(ND$_{SedLaunder}$) + 3*(W$_{Sed}$/2 - outerradius(ND$_{SedLaunder}$))/4
- z: tank$_{origin2}$ + W$_{Sed}$/2*tan(AN$_{SedBottom}$) + ElbowRadius(ND$_{SedPipeEntrance}$)

"y" - specifies axis that object will be rotated about.



Top View

rotate2a - Rotate$_{3d}$ turns the object based on a given axis and degree angle.

rotate2a - rotate$_{3d}$(p1,sedtankpipe1$_{origin}$,"z",sedtankpipe5$_{angle}$)

sedtankpipe1$_{origin}$ =

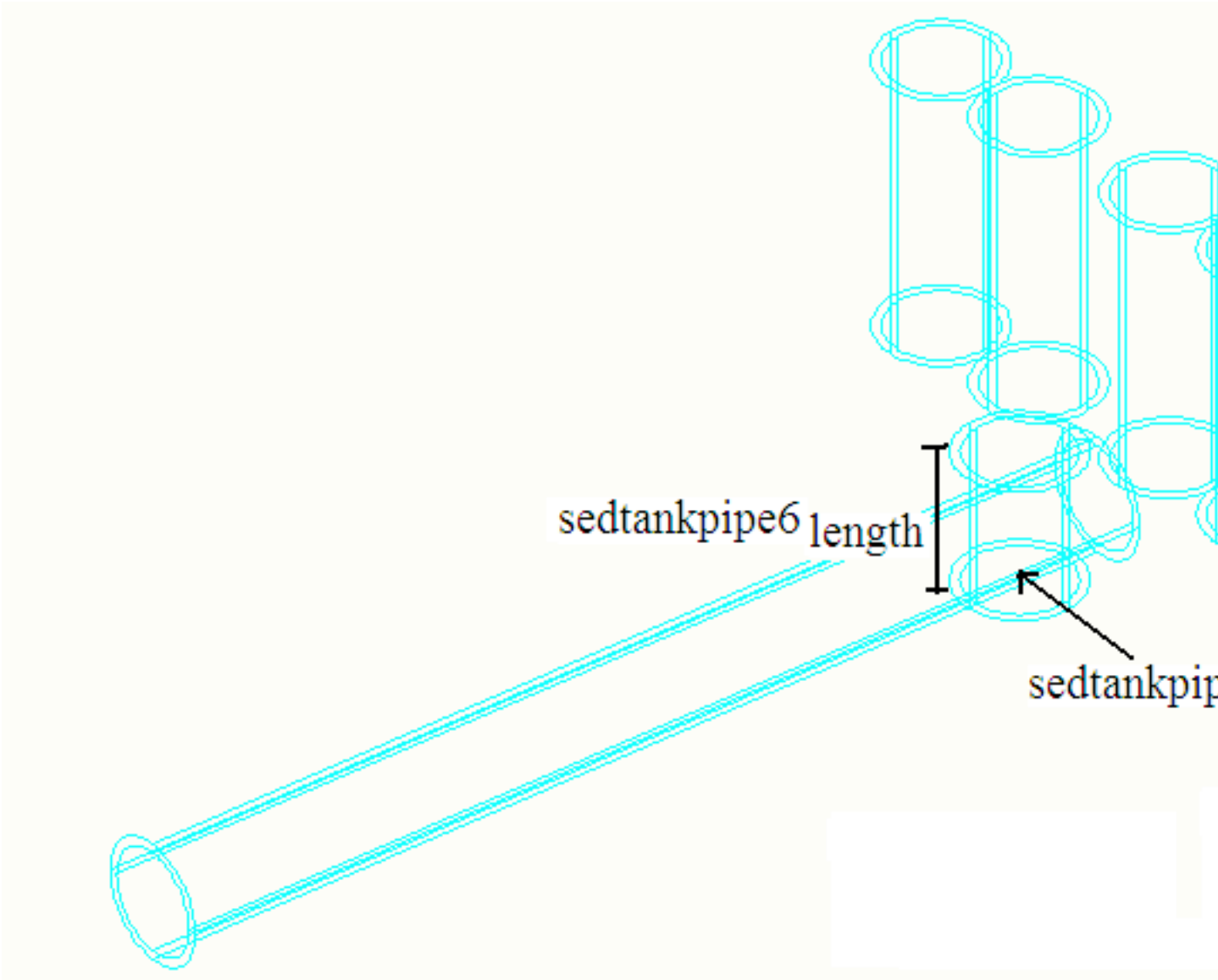
- x: tank$_{origin0}$ - L$_{Sed}$ + W$_{Channel}$/2
- y: tank$_{origin1}$ + W$_{Sed}$/2 + outerradius(ND$_{SedLaunder}$) + 3*(W$_{Sed}$/2 - outerradius(ND$_{SedLaunder}$))/4
- z: tank$_{origin2}$ + H$_{Sed}$ - H$_{Channel}$

p1 =

- x: sedtankpipe5$_{origin0}$
- y: sedtankpipe5$_{origin1}$ + innerD(ND$_{SedPipeEntrance}$,EN$_{PipeSpec}$)/2
- z: sedtankpipe5$_{origin2}$ + innerD(ND$_{SedPipeEntrance}$,EN$_{PipeSpec}$)/2

"z" - specifies axis that object will be rotated about.

sedtankpipe5$_{angle}$ = -(rad/deg) * atan((((5*L$_{Sed}$)/8 - W$_{Channel}$/2)/ [3 * (W$_{Sed}$/2 - outerradius(ND$_{SedLaunder}$))/4) + outerradius(ND$_{SedLaunder}$))$^{-1}$)



Northeast Isometric View

pipe6 - Calls the Pipe Program to create a pipe.

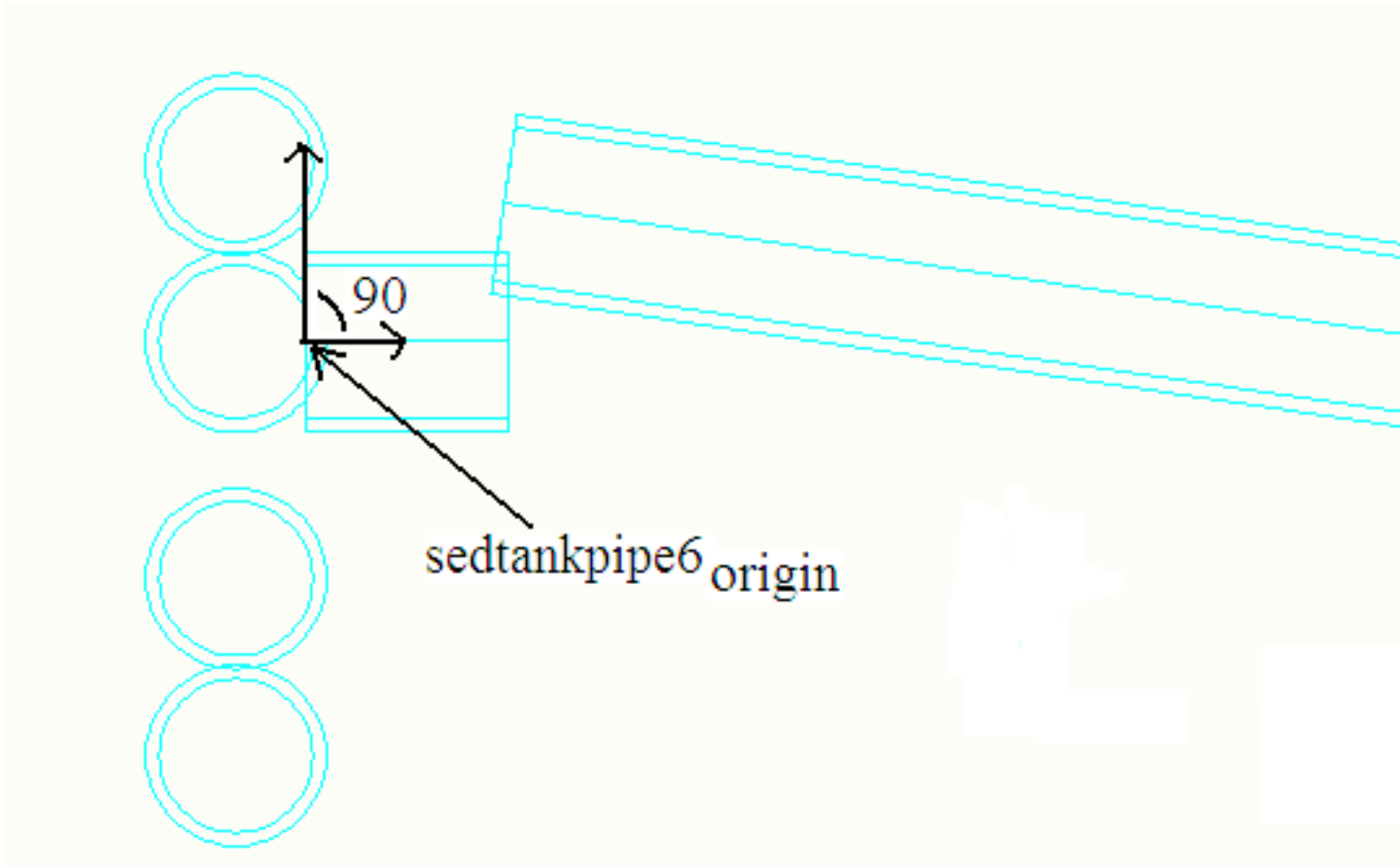
pipe6 <- Pipe($sedtankpipe6_{origin}$,$ND_{SedPipeEntrance}$,$sedtankpipe6_{length}$,$EN_{PipeSpec}$)

$sedtankpipe6_{origin}$ =

- x: $tank_{origin0}$ - L_{Sed} + $W_{Channel}/2$ + ElbowRadius($ND_{SedPipeEntrance}$)
- y: $tank_{origin1}$ + $W_{Sed}/2$ + outerradius($ND_{SedLaunder}$) + 1*($W_{Sed}/2$ - outerradius($ND_{SedLaunder}$))/4
- z: $tank_{origin2}$ + $W_{Sed}/2$*tan($AN_{SedBottom}$) + ElbowRadius($ND_{SedPipeEntrance}$)

$ND_{SedPipeEntrance}$ = Nominal diameter of entrance pipe from the inlet channel to the sedimentation tank.

$sedtankpipe6_{length}$ = $((L_{Sed}/8 - W_{InletChannel}/2)^2 + (\text{outerradius}(ND_{SedLaunder}) + 1*(W_{Sed}/2 - \text{outerradius}(ND_{SedLaunder}))/4)^2)^{1/2}$ - 2ElbowRadius($ND_{SedPipeEntrance}$)



Top View

rotate3 - $Rotate_{3d}$ turns the object based on a given axis and degree angle.

rotate3 <- $rotate_{3d}$(p1,$sedtankpipe6_{origin}$,"y",90)

p1 =

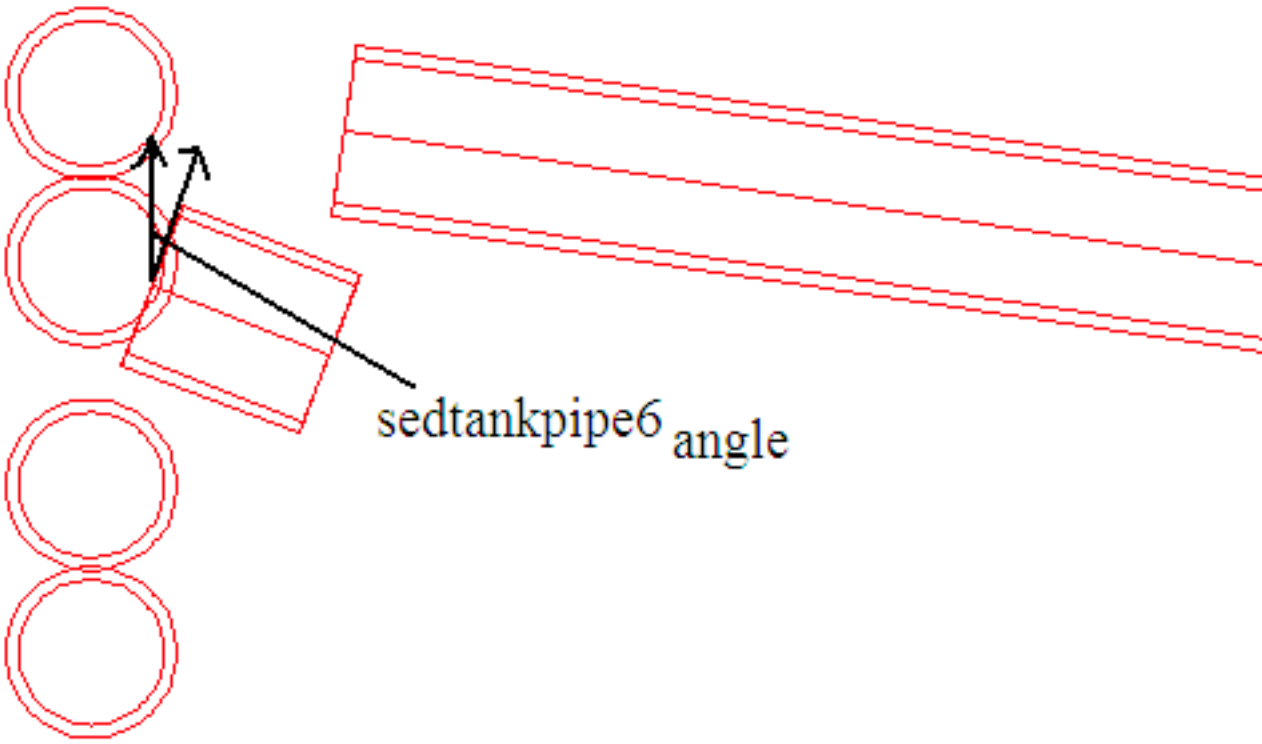
- x: $sedtankpipe_{origin0}$ + innerD($ND_{SedPipeEntrance}$,$EN_{PipeSpec}$)/2
- y: $sedtankpipe5_{origin1}$
- z: $sedtankpipe5_{origin0}$

$sedtankpipe6_{origin}$ =

- x: $tank_{origin0}$ - L_{Sed} + $W_{Channel}/2$ + ElbowRadius($ND_{SedPipeEntrance}$)
- y: $tank_{origin1}$ + $W_{Sed}/2$ + outerradius($ND_{SedLaunder}$) + 1*($W_{Sed}/2$ - outerradius($ND_{SedLaunder}$))/4
- z: $tank_{origin2}$ + $W_{Sed}/2$*tan($AN_{SedBottom}$) + ElbowRadius($ND_{SedPipeEntrance}$)

"y" - specifies axis that object will be rotated about.

90 - rotation angle



Top View

rotate3a - [Rotate$_{3d}$] turns the object based on a given axis and degree angle.

rotate3a <- rotate$_{3d}$(p1,sedtankpipe2$_{origin}$,"z",sedtankpipe6$_{angle}$)

p1 =

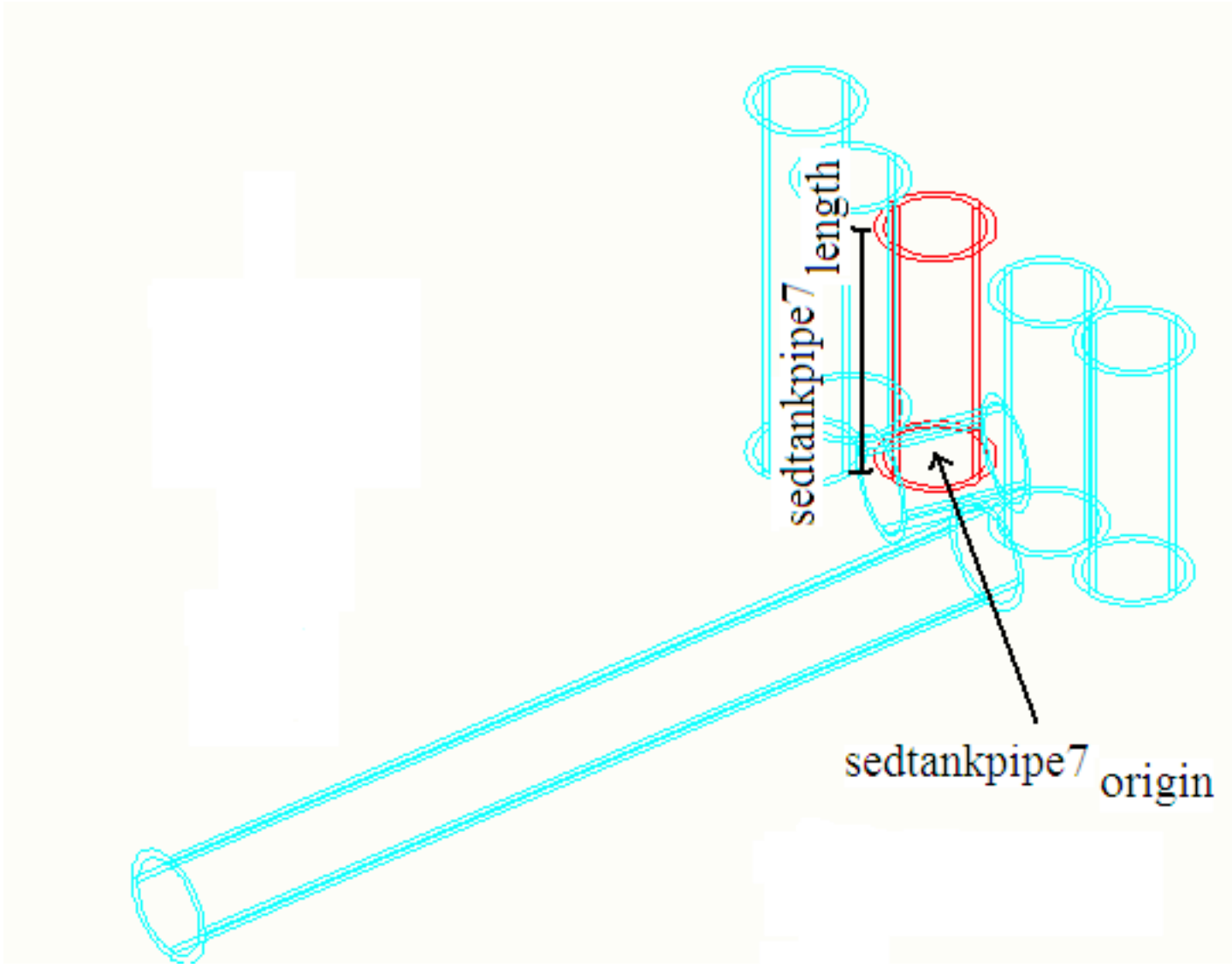
- x: sedtankpipe6$_{origin0}$
- y: sedtankpipe6$_{origin1}$ + innerD(ND$_{SedPipeEntrance}$,EN$_{PipeSpec}$)/2
- z: sedtankpipe6$_{origin2}$ + innerD(ND$_{SedPipeEntrance}$,EN$_{PipeSpec}$)/2

sedtankpipe2$_{origin}$ =

- x: tank$_{origin0}$ - L$_{Sed}$ + W$_{Channel}$/2
- y: tank$_{origin1}$ + W$_{Sed}$/2 + outerradius(ND$_{SedLaunder}$) + 1*(W$_{Sed}$/2 - outerradius(ND$_{SedLaunder}$))/4
- z: tank$_{origin2}$ + H$_{Sed}$ - H$_{Channel}$

"z" - specifies axis that object will be rotated about.

sedtankpipe6$_{angle}$ = -(rad/deg) * (((L$_{Sed}$/8 - W$_{Channel}$/2)/ ((W$_{Sed}$/2 - outerradius(ND$_{SedLaunder}$))/4) + outerradius(ND$_{SedLaunder}$))$^{-1}$)



NorthEast Isometric View

pipe7 - Calls the Pipe Program to create a pipe.

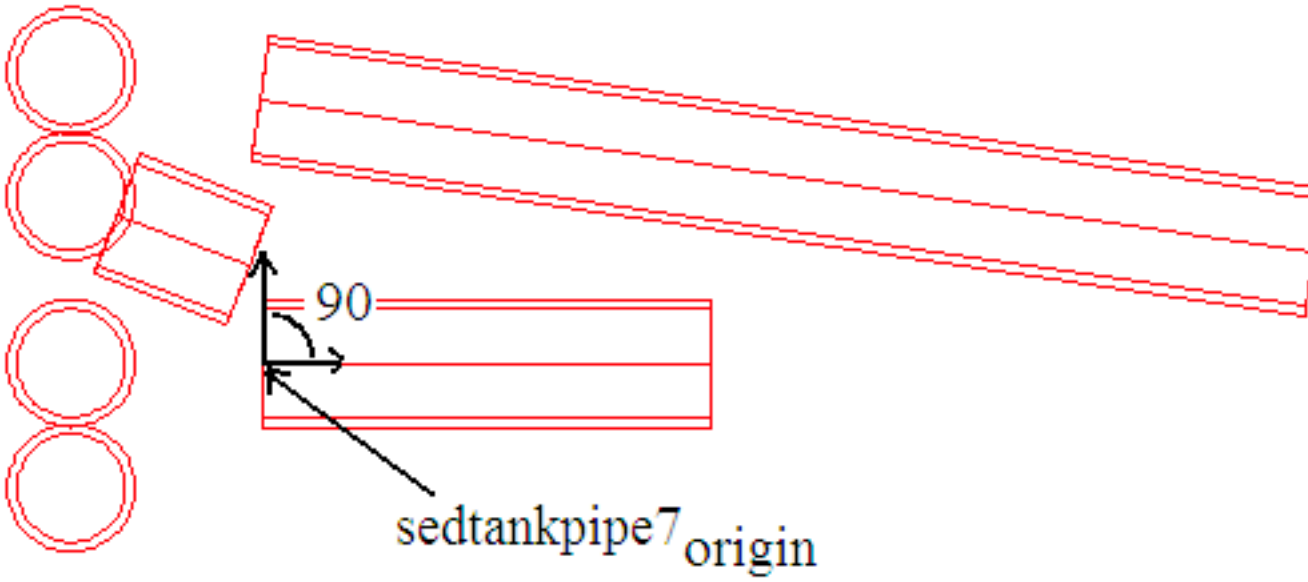
pipe7 <- Pipe($sedtank7_{origin}$,$ND_{SedPipeEntrance}$,$sedtankpipe7_{length}$,$EN_{PipeSpec}$)

$sedtankpipe7_{origin}$ =

- x: $tank_{origin0}$ - L_{Sed} + $W_{Channel}/2$ + ElbowRadius($ND_{SedPipeEntrance}$)
- y: $tank_{origin1}$ + $W_{Sed}/2$ - outerradius($ND_{SedLaunder}$) - 1*($W_{Sed}/2$ - outerradius($ND_{SedLaunder}$))/4
- z: $tank_{origin2}$ + $W_{Sed}/2$*tan($AN_{SedBottom}$) + ElbowRadius($ND_{SedPipeEntrance}$)

$sedtankpipe7_{length}$ = $((3L_{Sed}/8 - W_{InletChannel}/2)^2 + (\text{outerradius}(ND_{SedLaunder}) - 1*W_{Sed}/2 - \text{outerradius}(ND_{SedLaunder}))/4)^2)^{1/2}$ - 2ElbowRadius($ND_{SedPipeEntrance}$)

$EN_{PipeSpec}$ = Enumerated type



Top View

rotate4 - [Rotate$_{3d}$] turns the object based on a given axis and degree angle.

rotate4 <- rotate$_{3d}$(p1,sedtankpipe7$_{origin}$,"y",90)

p1 =

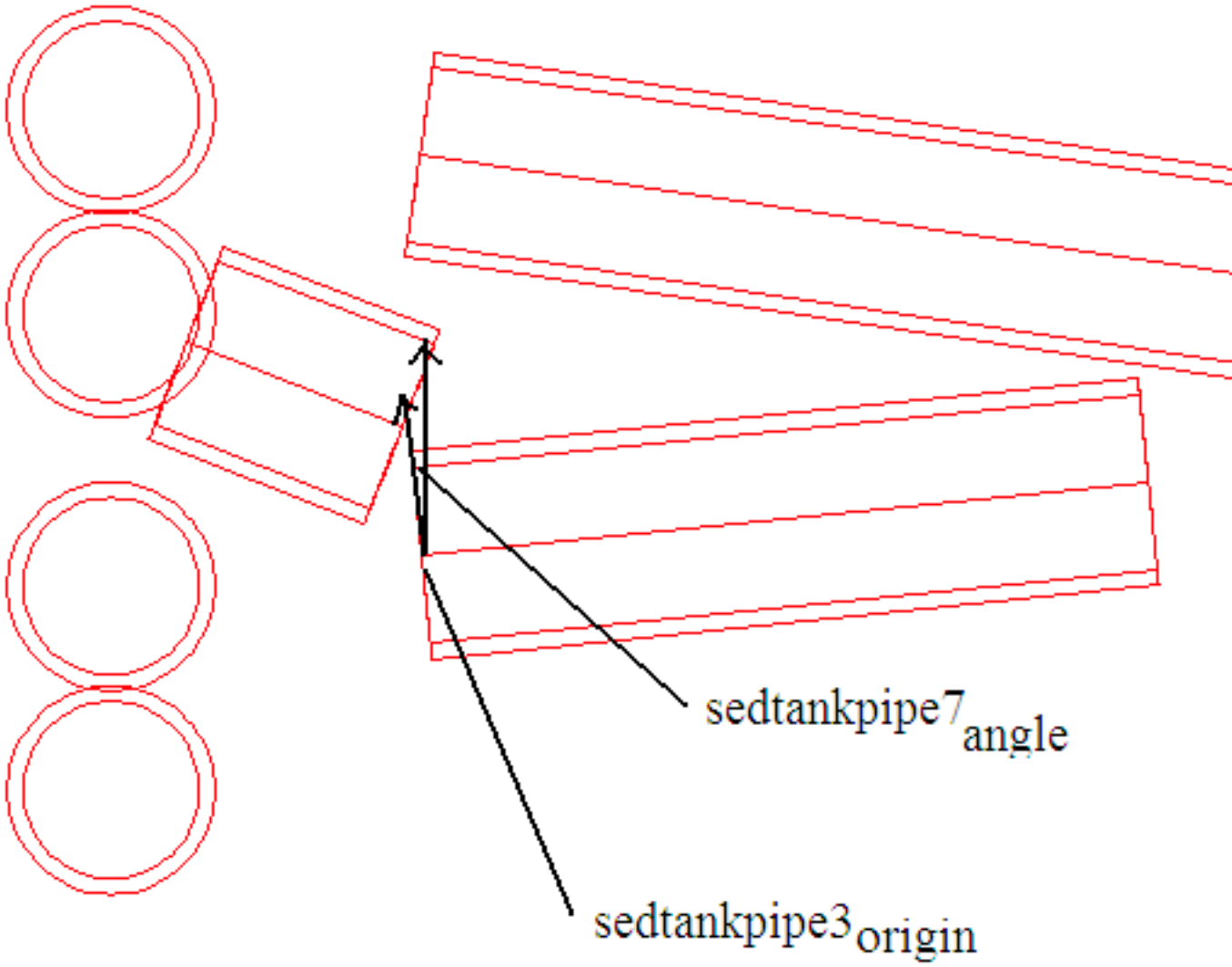
- x: sedtankpipe7$_{origin0}$ + innerD(ND$_{SedPipeEntrance}$,EN$_{PipeSpec}$)/2
- y: sedtankpipe7$_{origin1}$
- z: sedtankpipe7$_{origin2}$

sedtankpipe7$_{origin}$ =

- x: tank$_{origin0}$ - L$_{Sed}$ + W$_{Channel}$/2 + ElbowRadius(ND$_{SedPipeEntrance}$)
- y: tank$_{origin1}$ + W$_{Sed}$/2 - outerradius(ND$_{SedLaunder}$) - 1*(W$_{Sed}$/2 - outerradius(ND$_{SedLaunder}$))/4
- z: tank$_{origin2}$ + W$_{Sed}$/2*tan(AN$_{SedBottom}$) + ElbowRadius(ND$_{SedPipeEntrance}$)

"y" - specifies axis that object will be rotated about.

90 - Specifies rotation angle.



Top View

rotate4a - Rotate$_{3d}$ turns the object based on a given axis and degree angle.

rotate4a <- rotate$_{3d}$(p1,sedtankpipe3$_{origin}$,"z",sedtankpipe7$_{angle}$)

p1 =

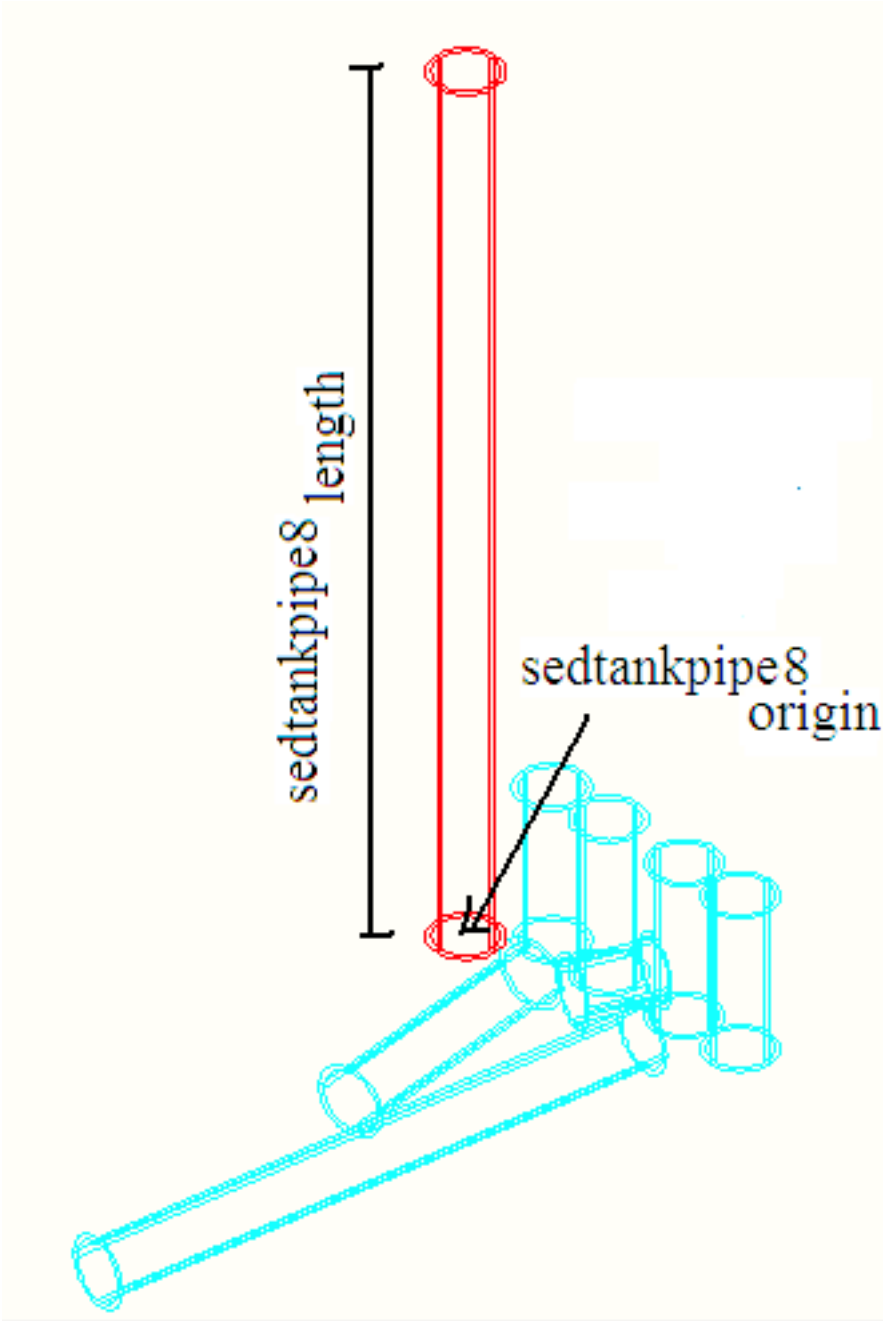
- x: sedtankpipe7$_{origin0}$
- y: sedtankpipe7$_{origin1}$ + innerD(ND$_{SedPipeEntrance}$,EN$_{PipeSpec}$)/2
- z: sedtankpipe7$_{origin2}$ + innerD(ND$_{SedPipeEntrance}$,EN$_{PipeSpec}$)/2

sedtankpipe3$_{origin}$ =

- x: tank$_{origin0}$ - L$_{Sed}$ + W$_{Channel}$/2
- y: tank$_{origin1}$ + W$_{Sed}$/2 - outerradius(ND$_{SedLaunder}$) - 1*(W$_{Sed}$/2 - outerradius(ND$_{SedLaunder}$))/4
- z: tank$_{origin2}$ + H$_{Sed}$ - H$_{Channel}$

"z" - specifies axis that object will be rotated about.

sedtankpipe7$_{angle}$ = -(rad/deg) * atan(((3*L_{Sed}/8 - $W_{Channel}$/2)/ ((W_{Sed}/2 - outerradius($ND_{SedLaunder}$))/4) + outerradius($ND_{SedLaunder}$))$^{-1}$)



Northeast Isometric View

pipe8 - Calls the Pipe Program to create a pipe.

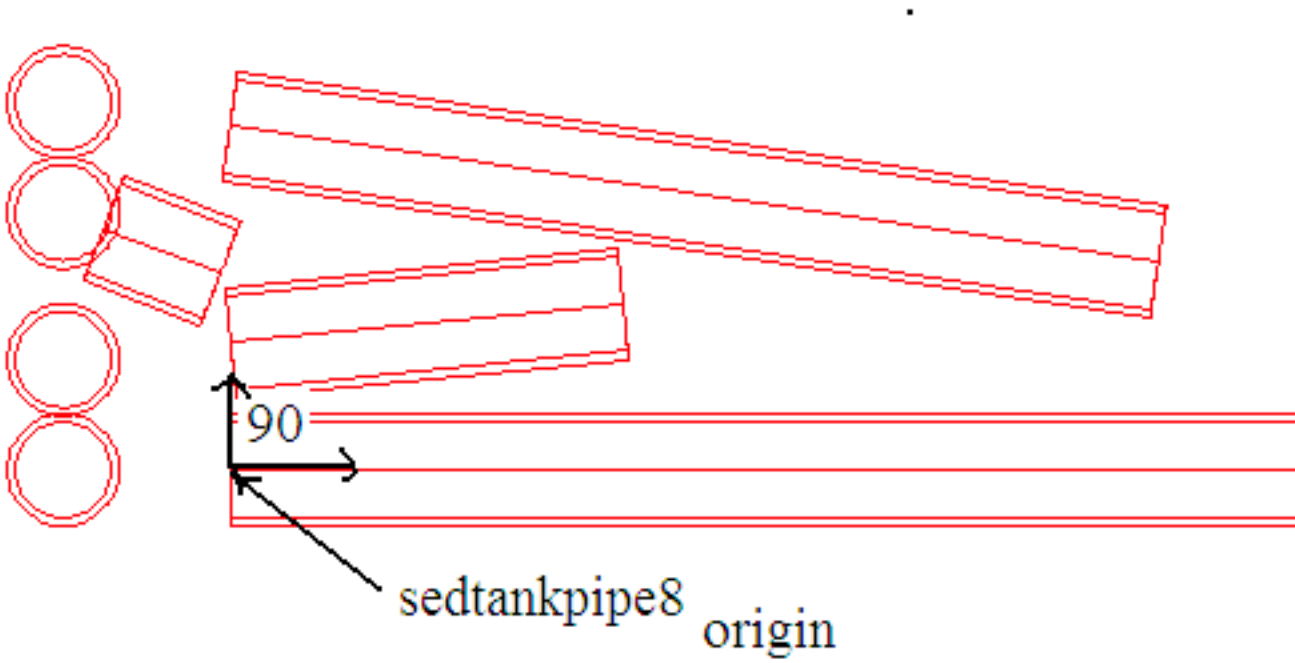
pipe8 <- Pipe(sedtankpipe8$_{origin}$,$ND_{SedPipeEntrance}$,sedtankpipe8$_{length}$,$EN_{PipeSpec}$)

sedtankpipe8$_{origin}$ =

- x: tank$_{origin0}$ - L_{Sed} + $W_{Channel}$/2 + ElbowRadius($ND_{SedPipeEntrance}$)
- y: tank$_{origin1}$ + W_{Sed}/2 - outerradius($ND_{SedLaunder}$) - 3*(W_{Sed}/2 - outerradius($ND_{SedLaunder}$))/4
- z: tank$_{origin2}$ + W_{Sed}/2*tan($AN_{SedBottom}$) + ElbowRadius($ND_{SedPipeEntrance}$)

$ND_{SedPipeEntrance}$ = Nominal diameter of entrance pipe from the inlet channel to the sedimentation tank.

sedtankpipe8$_{length}$ = ((7L_{Sed}/8 - $W_{InletChannel}$/2)^2 + (outerradius($ND_{SedLaunder}$) - 3*W_{Sed}/2 - outerradius($ND_{SedLaunder}$))/4)^2)$^{1/2}$ - 2ElbowRadius($ND_{SedPipeEntrance}$)



Top View

rotate5 - [Rotate$_{3d}$] turns the object based on a given axis and degree angle.

rotate5 <- rotate$_{3d}$(p1,sedtankpipe8$_{origin}$,"y",90)

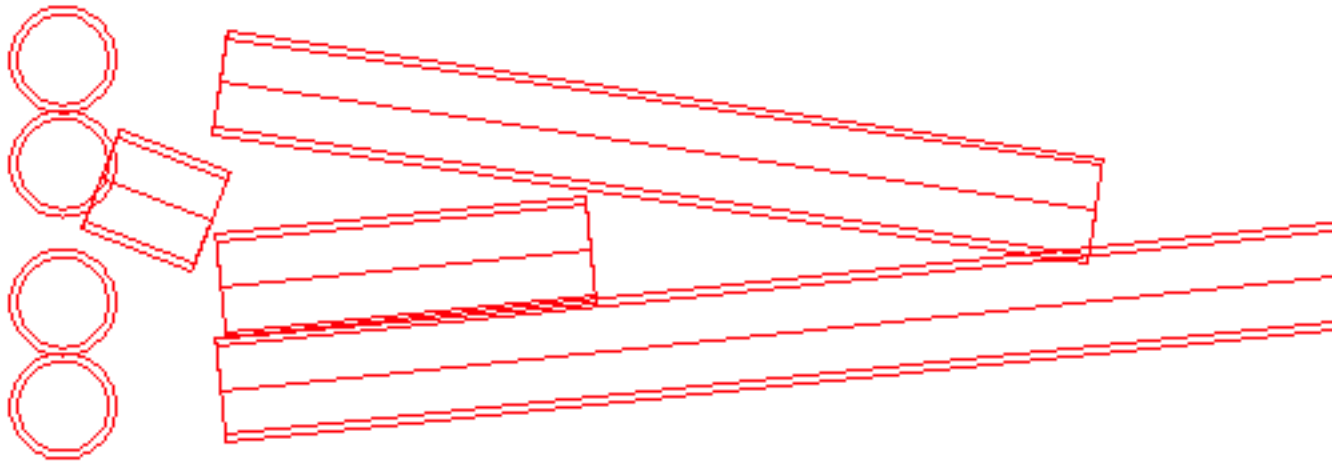
p1 =

- x: sedtankpipe8$_{origin0}$ + innerD($ND_{SedPipeEntrance}$,$EN_{PipeSpec}$)/2
- y: sedtankpipe8$_{origin1}$
- z: sedtankpipe8$_{origin2}$

sedtankpipe8$_{origin}$ =

- x: tank$_{origin0}$ - L_{Sed} + $W_{Channel}$/2 + ElbowRadius($ND_{SedPipeEntrance}$)
- y: tank$_{origin1}$ + W_{Sed}/2 - outerradius($ND_{SedLaunder}$) - 3*(W_{Sed}/2 - outerradius($ND_{SedLaunder}$))/4
- z: tank$_{origin2}$ + W_{Sed}/2*tan($AN_{SedBottom}$) + ElbowRadius($ND_{SedPipeEntrance}$)

"y" - specifies axis that object will be rotated about.



Top View

rotate5a - Rotate$_{3d}$ turns the object based on a given axis and degree angle.

rotate5a <- rotate$_{3d}$(p1,sedtankpipe4$_{origin}$,"z",sedtankpipe8$_{angle}$)

p1 =

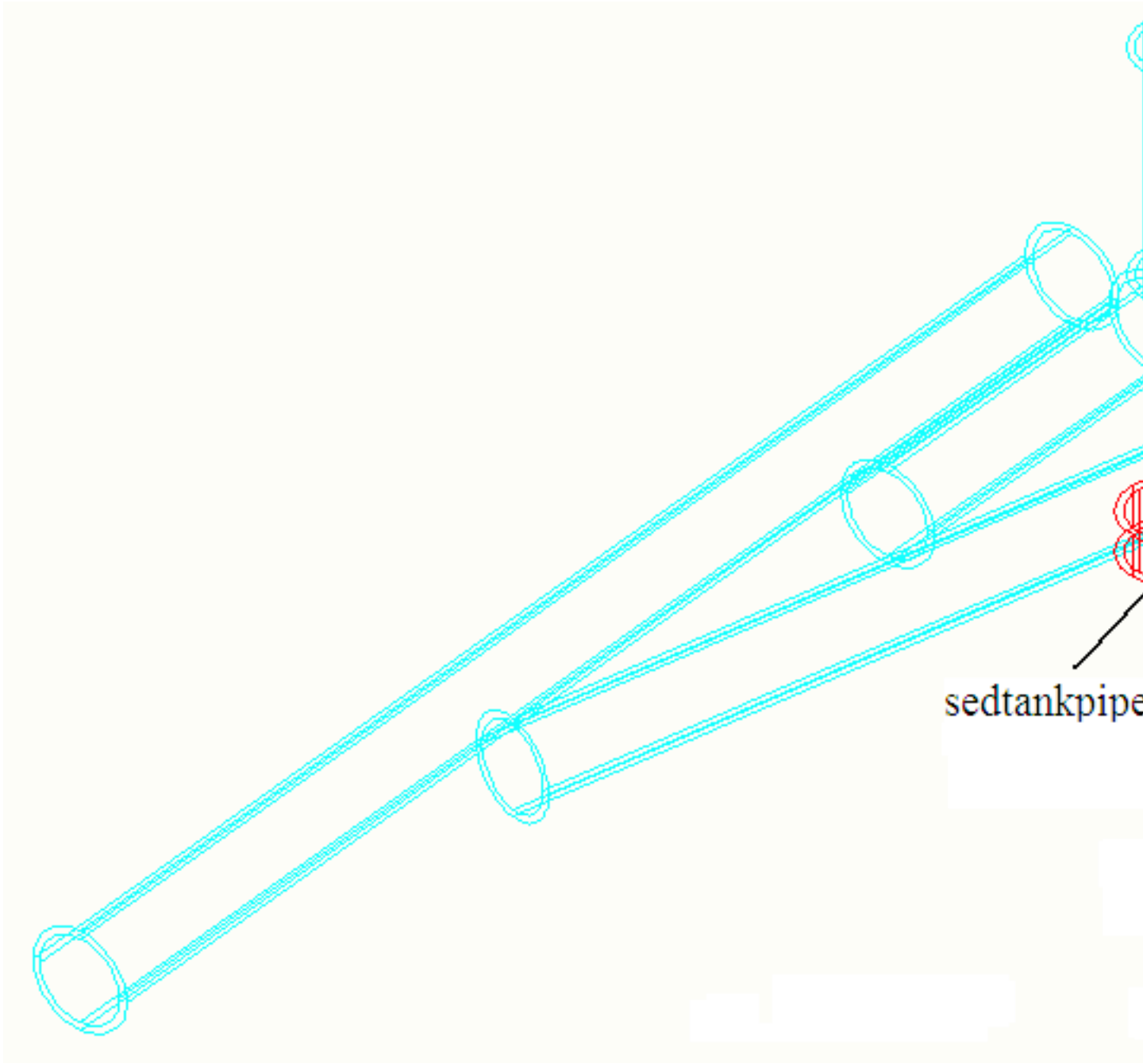
- x: sedtankpipe8$_{origin0}$
- y: sedtankpipe8$_{origin1}$ + innerD(ND$_{SedPipeEntrance}$,EN$_{PipeSpec}$)/2
- z: sedtankpipe8$_{origin2}$ + innerD(ND$_{SedPipeEntrance}$,EN$_{PipeSpec}$)/2

sedtankpipe4$_{origin}$ =

- x: tank$_{origin0}$ - L$_{Sed}$ + W$_{Channel}$/2
- y: tank$_{origin1}$ + W$_{Sed}$/2 - outerradius(ND$_{SedLaunder}$) - 3*(W$_{Sed}$/2 - outerradius(ND$_{SedLaunder}$))/4
- z: tank$_{origin2}$ + H$_{Sed}$ - H$_{Channel}$

"z" - specifies axis that object will be rotated about.

sedtankpipe8$_{angle}$ = -(rad/deg) * atan((((7*L$_{Sed}$)/8 - W$_{Channel}$/2)/ (3*(W$_{Sed}$/2 - outerradius(ND$_{SedLaunder}$))/4) + outerradius(ND$_{SedLaunder}$))$^{-1}$)



Northeast Isometric View

pipe9 - Calls the Pipe Program to create a pipe.

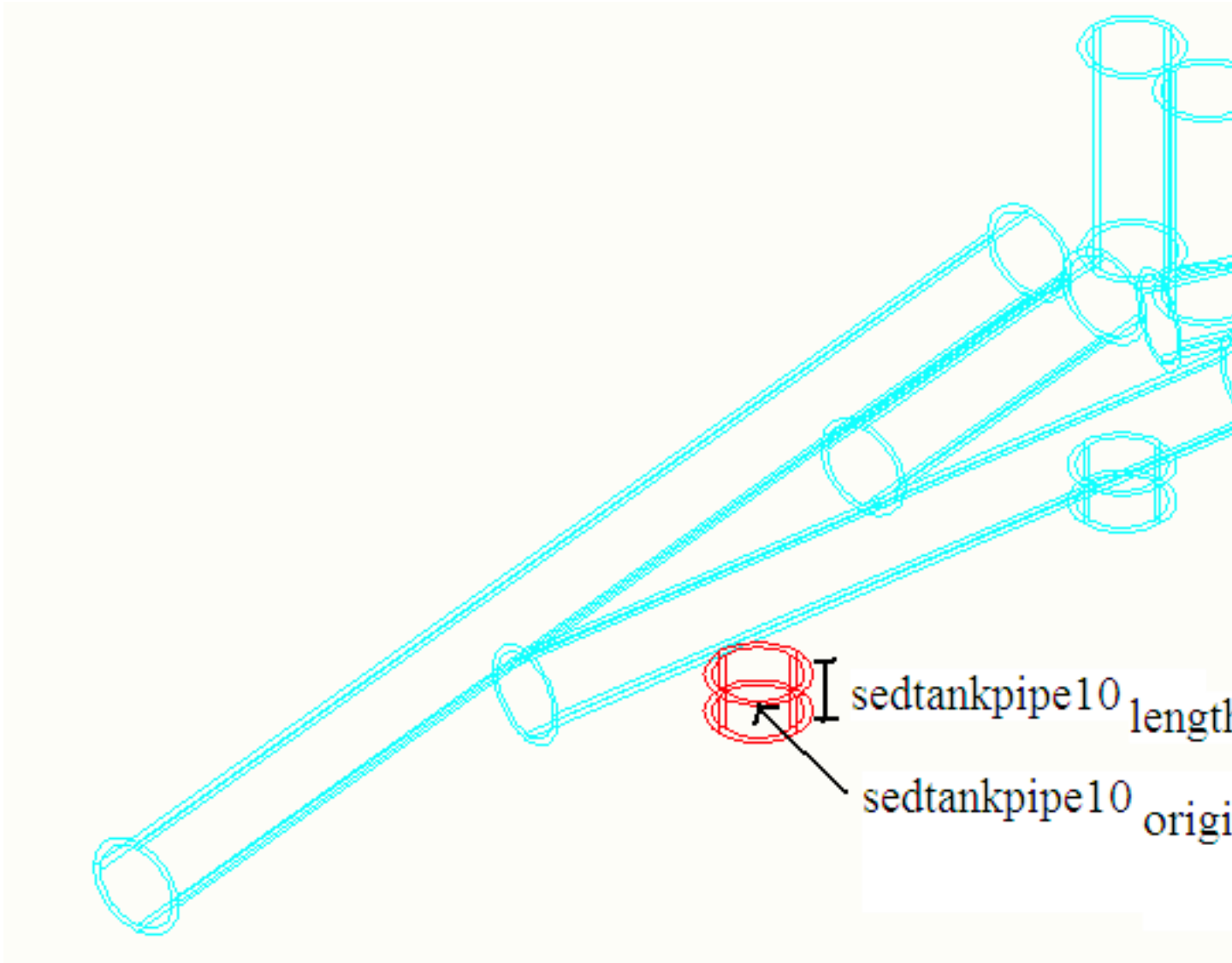
pipe9 <- Pipe($sedtankpipe9_{origin}$,$ND_{SedPipeEntrance}$,$sedtankpipe9_{length}$,$EN_{PipeSpec}$)

$sedtankpipe9_{origin}$ =

- x: $tank_{origin0}$ - (7*L_{Sed})/8
- y: $tank_{origin1}$ + W_{Sed}/2
- z: $tank_{origin2}$ + $S_{SedInlet}$

$ND_{SedPipeEntrance}$ = Nominal diameter of entrance pipe from the inlet channel to the sedimentation tank

$sedtankpipe9_{length}$ = W_{Sed}/2 * tan($AN_{SedBottom}$) - ElbowRadius($ND_{SedPipeEntrance}$) - $S_{SedInlet}$ + ElbowRadius($ND_{SedPipeEntrance}$)



Northeast Isometric View

pipe10 - Calls the Pipe Program to create a pipe.

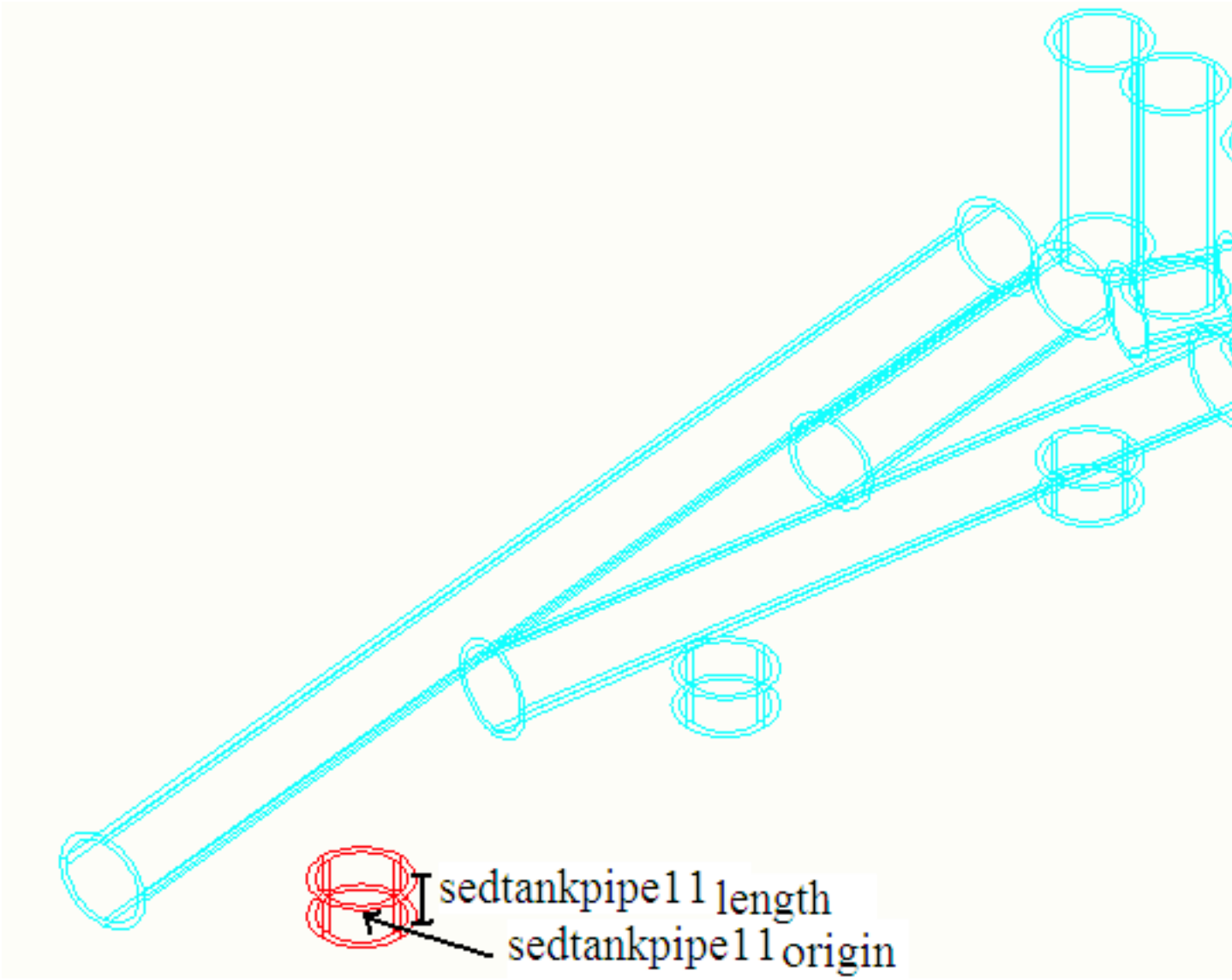
pipe10 <- Pipe($sedtankpipe10_{origin}$,$ND_{SedPipeEntrance}$,$sedtankpipe9_{length}$,$EN_{PipeSpec}$)

$sedtankpipe10_{origin}$ =

- x: $tank_{origin0}$ - (5*L_{Sed})/8
- y: $tank_{origin1}$ + W_{Sed}/2
- z: $tank_{origin2}$ + $S_{SedInlet}$

$ND_{SedPipeEntrance}$ = Nominal diameter of entrance pipe from the inlet channel to the sedimentation tank

$sedtankpipe9_{length}$ = W_{Sed}/2 * tan($AN_{SedBottom}$) - ElbowRadius($ND_{SedPipeEntrance}$) - $S_{SedInlet}$ + ElbowRadius($ND_{SedPipeEntrance}$)



Northeast Isometric View

pipe11 - Calls the Pipe Program to create a pipe.

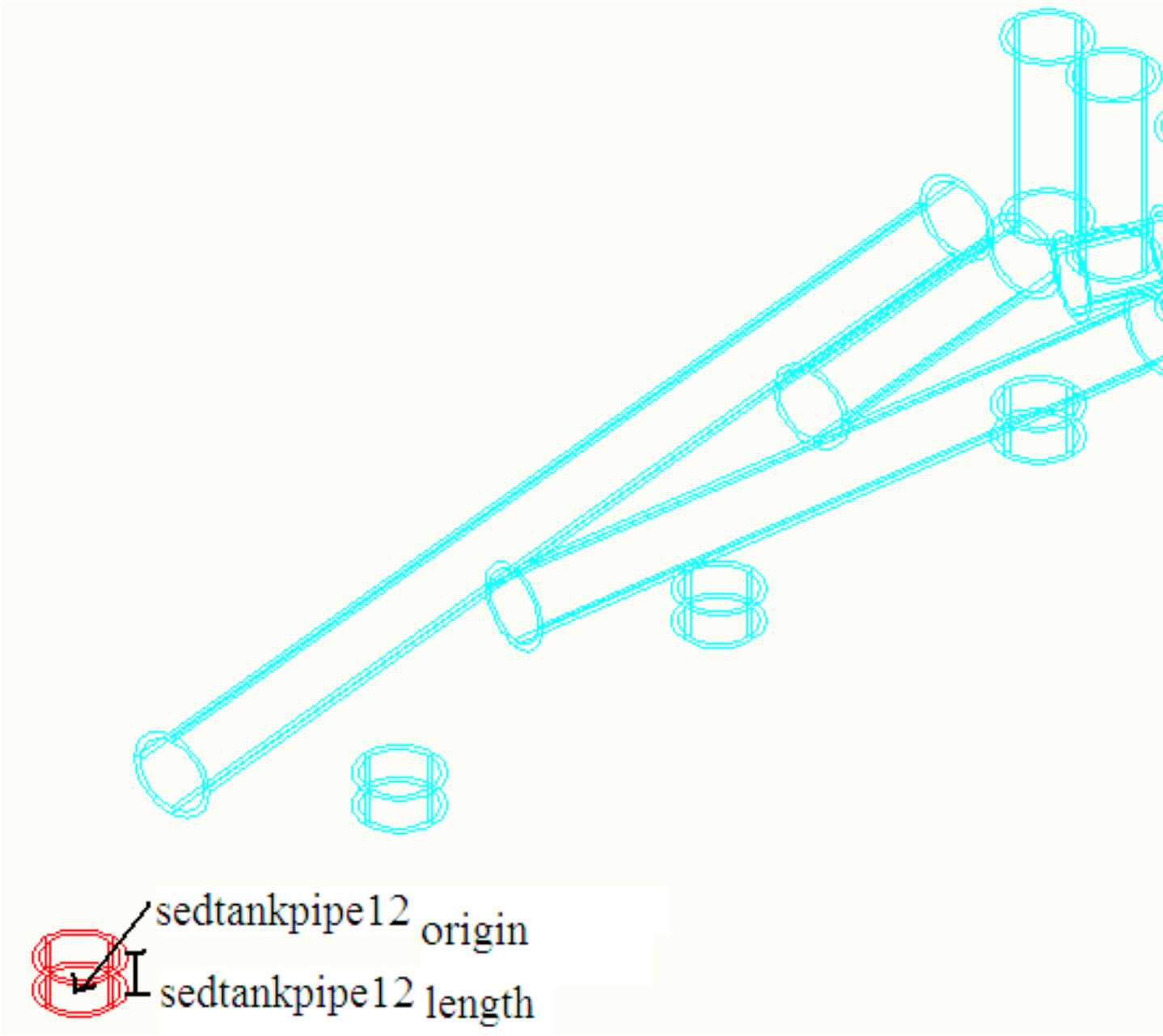
pipe11 <- Pipe($sedtankpipe11_{origin}$,$ND_{SedPipeEntrance}$,$sedtankpipe9_{length}$,$EN_{PipeSpec}$)

$sedtankpipe11_{origin}$ =

- x: $tank_{origin0}$ - (3*L_{Sed})/8
- y: $tank_{origin1}$ + W_{Sed}/2
- z: $tank_{origin2}$ + $S_{SedInlet}$

$ND_{SedPipeEntrance}$ = Nominal diameter of entrance pipe from the inlet channel to the sedimentation tank

$sedtankpipe9_{length}$ = W_{Sed}/2 * tan($AN_{SedBottom}$) - ElbowRadius($ND_{SedPipeEntrance}$) - $S_{SedInlet}$ + ElbowRadius($ND_{SedPipeEntrance}$)



Northeast Isometric View

pipe12 - Calls the Pipe Program to create a pipe.

pipe12 <- Pipe($sedtankpipe12_{origin}$,$ND_{SedPipeEntrance}$,$sedtankpipe9_{length}$,$EN_{PipeSpec}$)

$sedtankpipe12_{origin}$ =

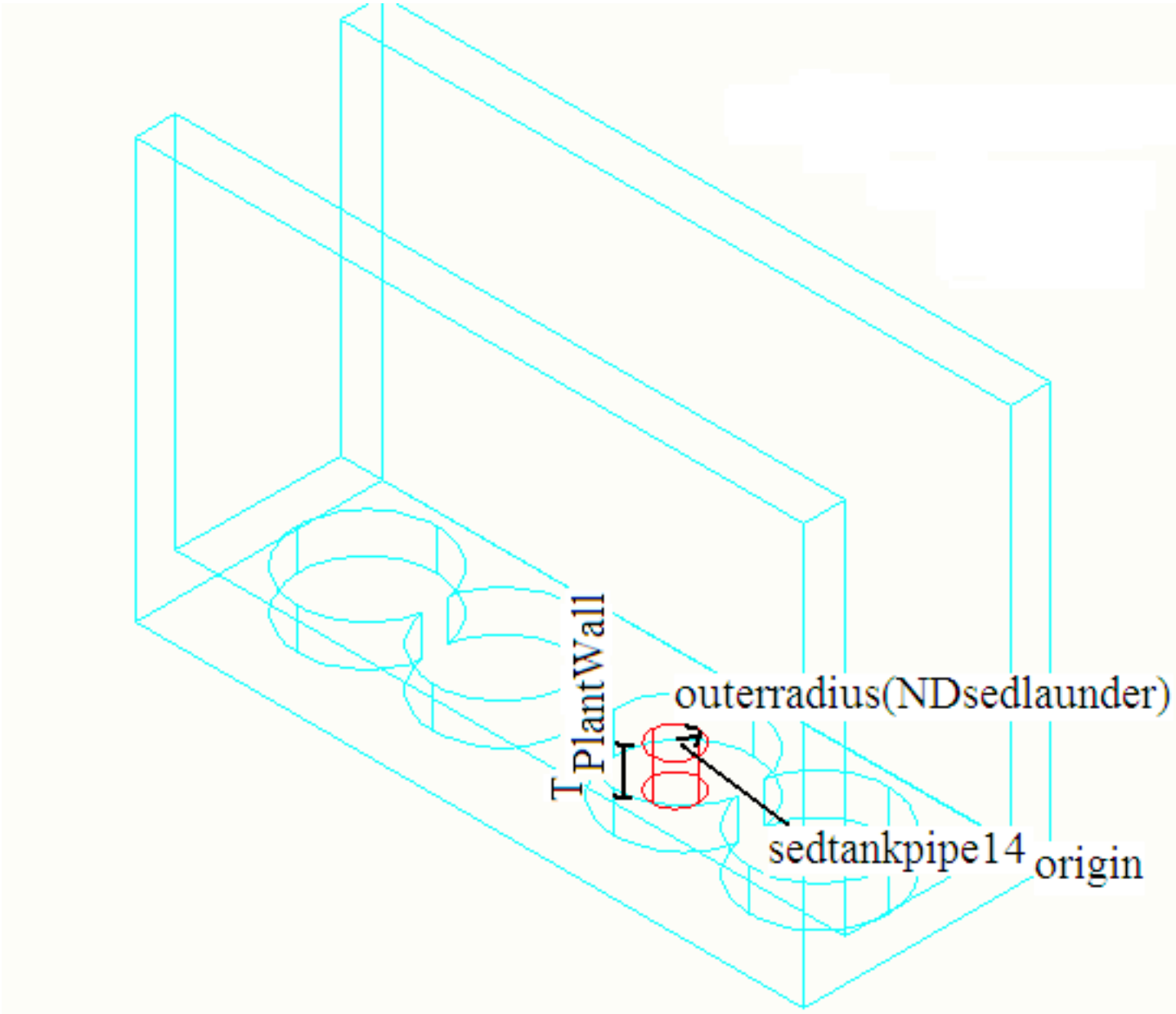
- x: $tank_{origin0}$ - (L_{Sed})/8
- y: $tank_{origin1}$ + W_{Sed}/2
- z: $tank_{origin2}$ + $S_{SedInlet}$

$ND_{SedPipeEntrance}$ = Nominal diameter of entrance pipe from the inlet channel to the sedimentation tank

$sedtankpip9_{length}$ = W_{Sed}/2 * tan($AN_{SedBottom}$) - ElbowRadius($ND_{SedPipeEntrance}$) - $S_{SedInlet}$ + ElbowRadius($ND_{SedPipeEntrance}$)

layerfreeze2 - $Layer_{freeze}$ locks the layer "pipe" so that it cannot be edited.

layerfreeze2 <- $layer_{freeze}$("pipe")



Northeast Isometric View

cylinder5 - CylinderA creates a cylinder based on three dimensions.

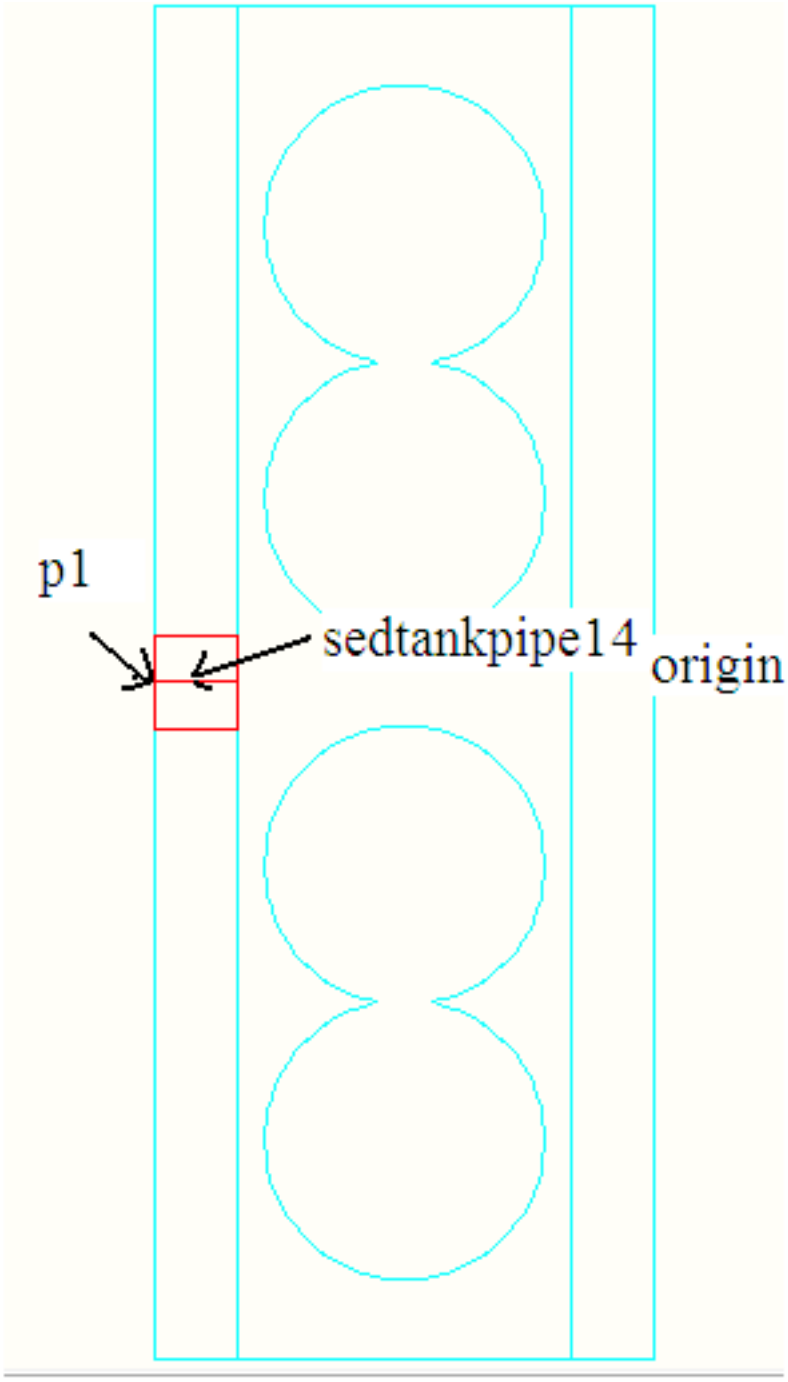
cylinder5 <- cylinderA(sedtankpipe14$_{origin}$,outerradius(ND$_{SedLaunder}$),-T$_{PlantWall}$)

sedtankpipe14$_{origin}$ =

- x: tank$_{origin0}$ - L$_{Sed}$
- y: tank$_{origin1}$ + W$_{Sed}$/2
- z: tank$_{origin2}$ + H$_{Sed}$ - H$_{Channel}$ - T$_{Mp}$

ND$_{SedLaunder}$ = Nominal diameter of launder.

T$_{ChannelWall}$ = Thickness of channel wall.



Top View

rotate2 - [Rotate$_{3d}$] turns the object based on a given axis and degree angle.

rotate2 <- rotate$_{3d}$(p1,sedtankpipe14$_{origin}$,"y",90)

p1 =

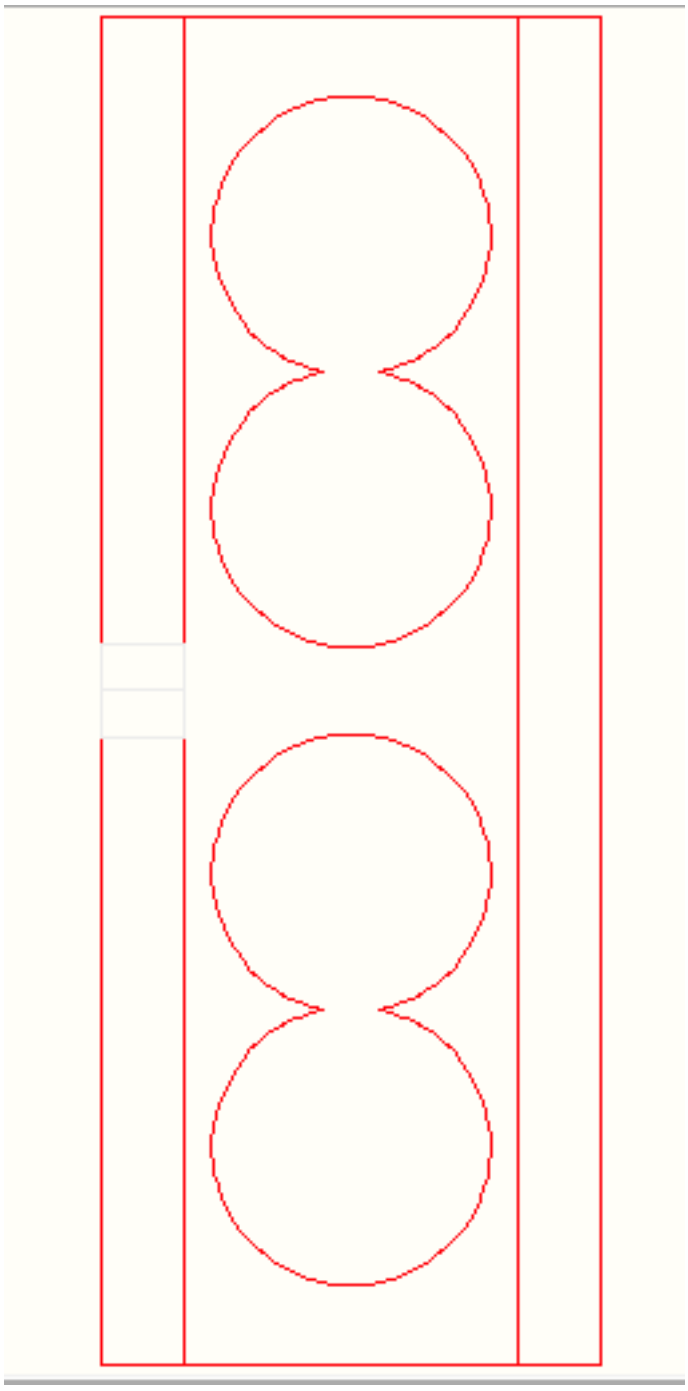
- x: sedtankpipe14$_{origin0}$ + outerradius(ND$_{SedLaunder}$)
- y: sedtankpipe14$_{origin1}$
- z: sedtankpipe14$_{origin2}$

sedtankpipe14$_{origin}$ =

- x: tank$_{origin0}$ - L$_{Sed}$
- y: tank$_{origin1}$ + W$_{Sed}$/2

- z: $tank_{origin2}$ + H_{Sed} - $H_{Channel}$ - T_{Mp}

"y" - specifies axis that object will be rotated about.



Top View

subtract2 - SubtractD subtracts a selected object based on two specified points.

subtract2 <- subtractD($Plant_{Origin}$,p1)

p1 =

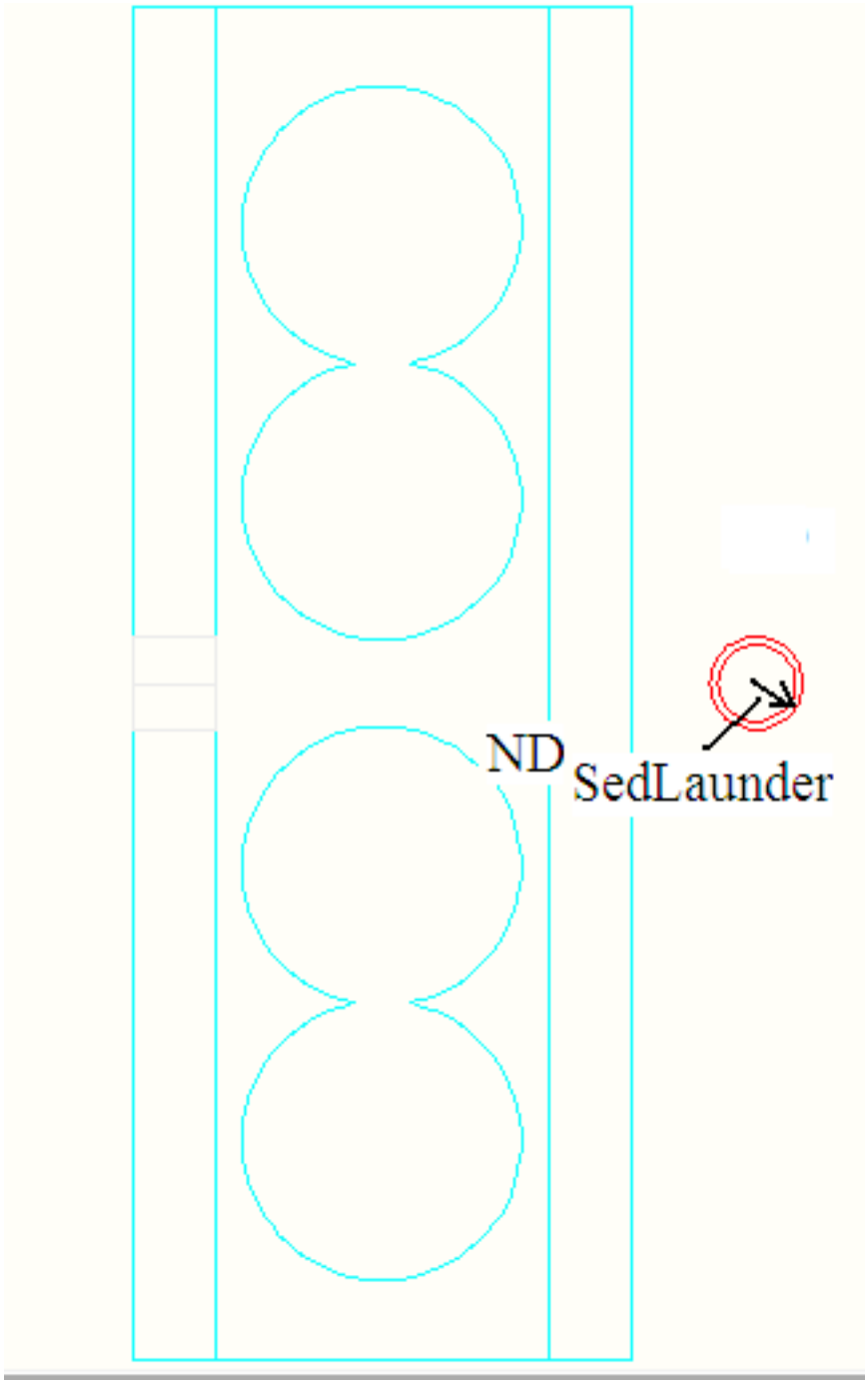
- x: $sedtankpipe14_{origin0}$
- y: $sedtankpipe14_{origin1}$ + outerradius($ND_{SedLaunder}$)
- z: $sedtankpipe14_{origin2}$
  //

//
layerthawp - Layer$_{thaw}$ unlocks the layer "pipe" so that it can be edited.

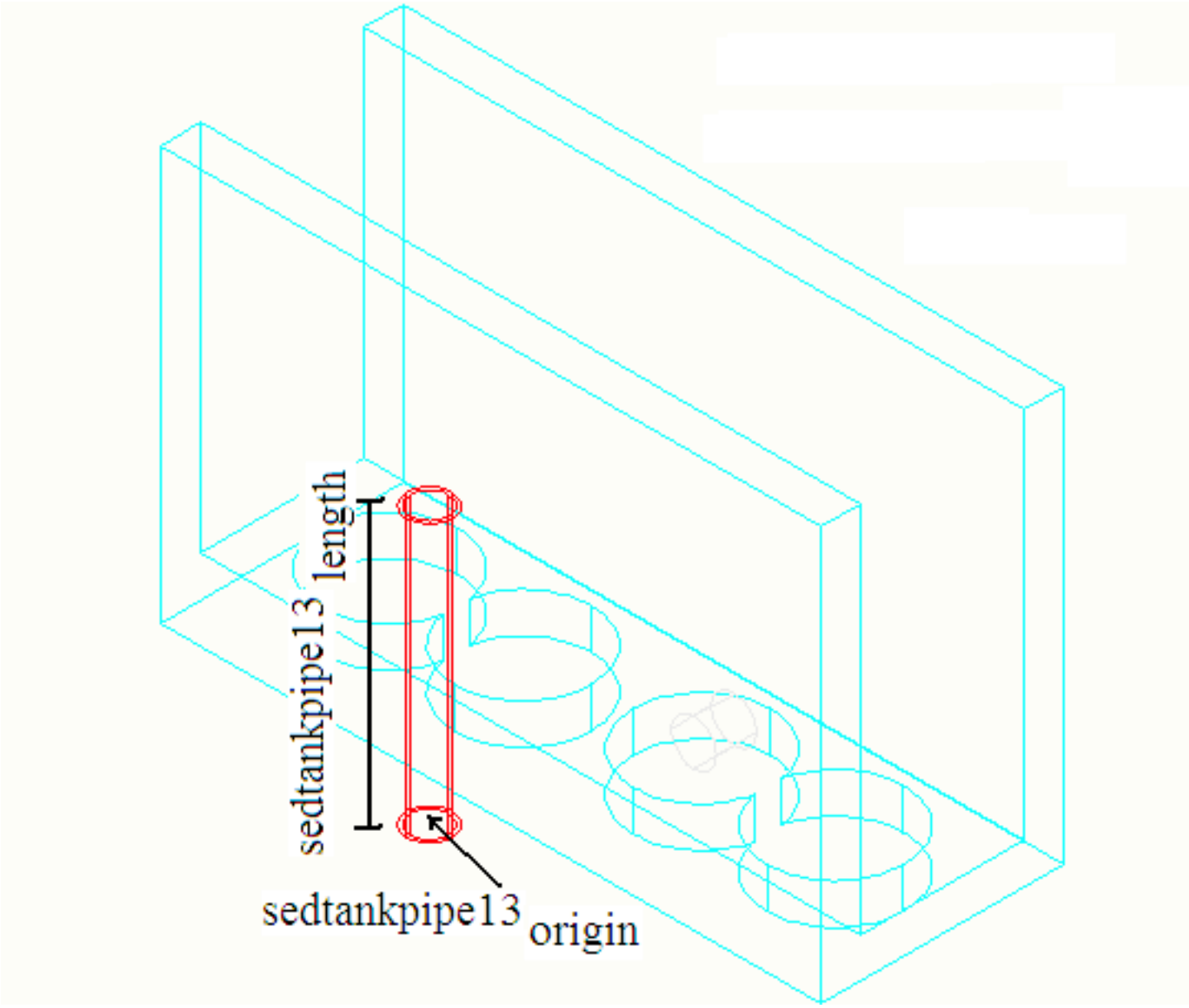
layerthawp <-layer$_{thaw}$("pipe")

layersetp - Layer$_{set}$ selects the layer "pipe".

layersetp <- layer$_{set}$("pipe")



Top View



NorthEast Isometric View

pipe13 - Calls the Pipe Program to create a pipe.

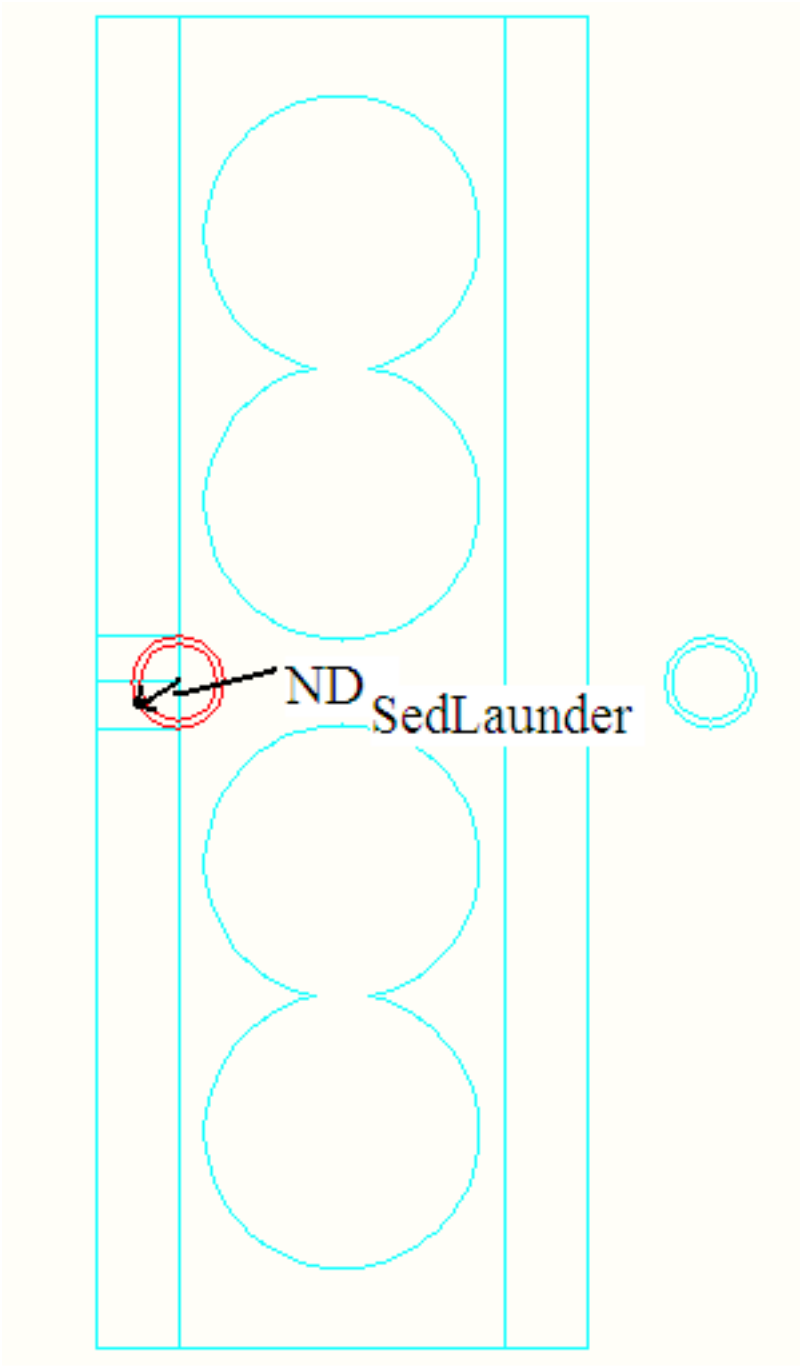
pipe13 <- Pipe($sedtankpipe13_{origin}$,$ND_{SedLaunder}$,$sedtankpipe13_{length}$,$EN_{PipeSpec}$)

$sedtankpipe13_{origin}$ =

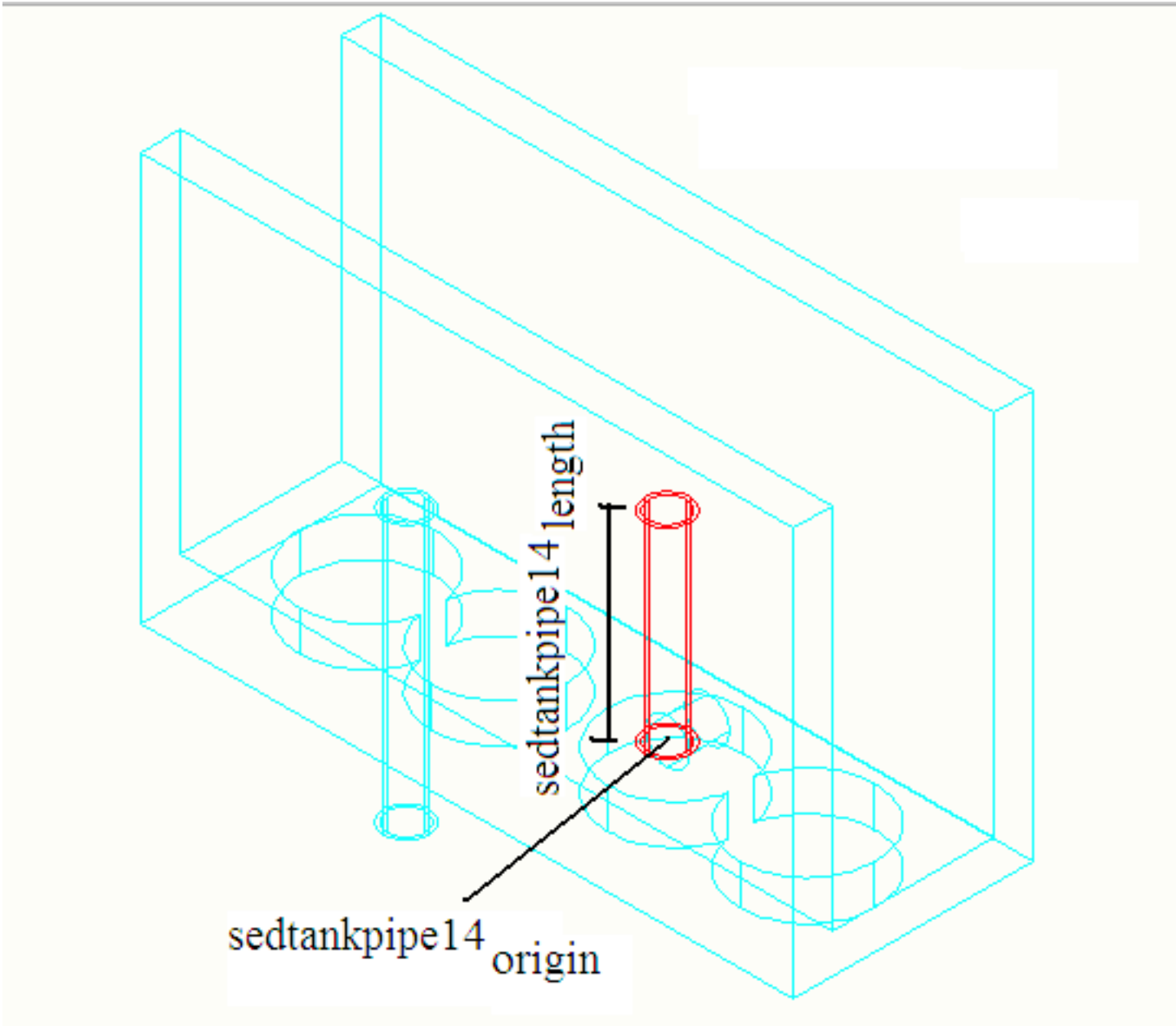
- x: $tank_{origin0}$ - L_{Sed} + $W_{Channel}$ + $T_{ChannelWall}$ + ElbowRadius($ND_{SedLaunder}$)
- y: $tank_{origin1}$ + $W_{Sed}/2$
- z: $tank_{origin2}$ + H_{Sed} - $H_{Channel}$ - T_{Mp} + ElbowRadius($ND_{SedLaunder}$)

$ND_{SedLaunder}$ = Nominal diameter of launder.

$sedtankpipe13_{length}$ = HW_{Sed} - $H_{SedAbove}/2$ - 2ElbowRadius($ND_{SedLaunder}$) - (H_{Sed} - $H_{InletChannel}$ - T_{Mp})



NorthEast Isometric View



Top View

pipe14 - Calls the Pipe Program to create a pipe.

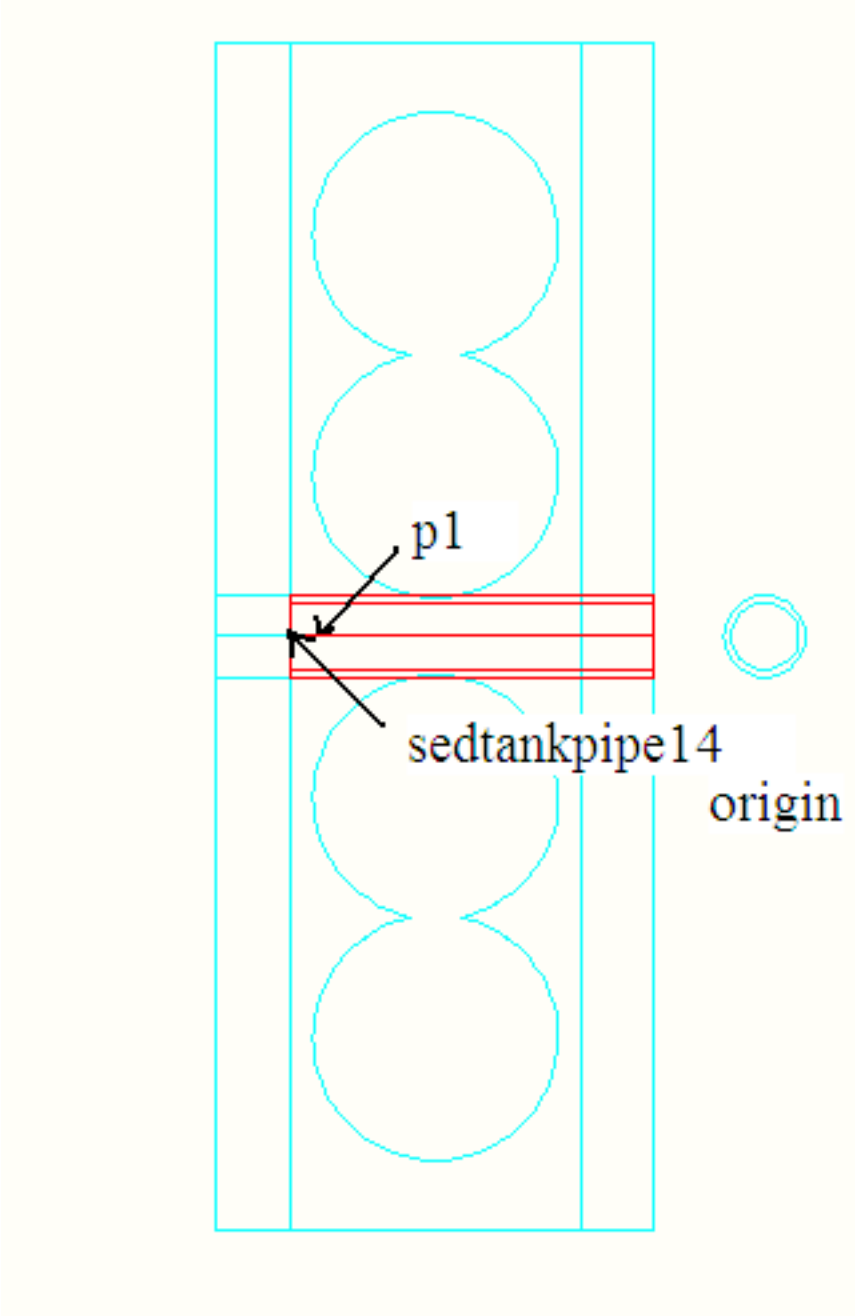
pipe14 <- Pipe($sedtankpipe14_{origin}$,$ND_{SedLaunder}$,$sedtankpipe14_{length}$,$EN_{PipeSpec}$)

$sedtankpipe14_{origin}$ =

- x: $tank_{origin0}$ - L_{Sed}
- y: $tank_{origin1}$ + $W_{Sed}/2$
- z: $tank_{origin2}$ + H_{Sed} - $H_{Channel}$ - T_{Mp}

$ND_{SedLaunder}$ = Nominal diameter of launder.

$sedtankpipe14_{length}$ = $W_{InletChannel}$ + $T_{ChannelWall}$



Top View

rotate6 - Rotate$_{3d}$ turns the object based on a given axis and degree angle.

rotate6 <- rotate$_{3d}$(p1,sedtankpipe14$_{origin}$,"y",90)

p1 =

- x: sedtankpipe14$_{origin0}$ + innerD(ND$_{SedLaunder}$,EN$_{PipeSpec}$)/2
- y: sedtankpipe14$_{origin1}$
- z: sedtankpipe14$_{origin2}$

sedtankpipe14$_{origin}$ =

- x: tank$_{origin0}$ - L$_{Sed}$
- y: tank$_{origin1}$ + W$_{Sed}$/2
- z: tank$_{origin2}$ + H$_{Sed}$ - H$_{Channel}$ - T$_{Mp}$

"y" - specifies axis that object will be rotated about.
layerfreeze3 - Layer$_{freeze}$ locks the layer "channel" so that it cannot be edited.

layerfreeze3 <- layer$_{freeze}$("pipe")

This page last changed on Nov 06, 2008 by ar329.

Inlet Pipe Drawing Script

layerthaw - $Layer_{thaw}$ unlocks the layer "channel" so that it can be edited.

layerthaw <-$layer_{thaw}$("channel")

viewtop - Rotates the object so that it is being viewed from the top.

viewtop <-viewtop1

cylinder1 - CylinderC is used to create a cylinder based on a point, radius and length.

cylinder1 <- cylinderC($sedtankinletpipe1_{origin}$,outerradius($ND_{SedPipeEntrance}$),-$sedtankinletpipe_{length}$)

$sedtankinletpipe1_{origin}$ =

- x: $Plant_{Origin0}$ - L_{Sed} + (.5$W_{InletChannel}$)
- y: $Plant_{Origin1}$ + (W_{Sed}/4)
- z: $Plant_{Origin2}$ + H_{Sed} - $H_{InletChannel}$

outerradius($ND_{SedPipeEntrance}$) = specifies the radius of the cylinder.

$sedtankinletpipe_{length}$ = specifies the length of the cylinder.

cylinder2 - CylinderC is used to create a cylinder based on a point, radius and length.

cylinder2 <- cylinderC($sedtankinletpipe2_{origin}$,outerradius($ND_{SedPipeEntrance}$),-$sedtankinletpipe_{length}$)

$sedtankinletpipe2_{origin}$ =

- x: $Plant_{Origin0}$ - L_{Sed} + (.5$W_{InletChannel}$)
- y: $Plant_{Origin1}$ + (3W_{Sed}/4)
- z: $Plant_{Origin2}$ + H_{Sed} - $H_{InletChannel}$

outerradius($ND_{SedPipeEntrance}$) = specifies the radius of the cylinder.

$sedtankinletpipe_{length}$ = specifies the length of the cylinder.

subtractc - SubtractC subtracts one object from the other based on three points.

subtractc <- subtractH($channel_{origin}$,p1,p2)

When sedlayout = 1, channelorigin is 3*1 matrix with x,y,z points corresponding to where the channel will be drawn. Channelorigin is called from the Channel Program.

Note: p1 and p2 are dummy variable used only in the program help section to designate the matrix below.

p1 =

- x: $sedtankinletpipe2_{origin0}$ + outerradius($ND_{SedPipeEntrance}$)
- y: $sedtankinletpipe2_{origin1}$ + outerradius($ND_{SedPipeEntrance}$)
- z: $sedtankpipe1_{origin2}$

p2 =

- x: $sedtankinletpipe1_{origin0}$ - outerradius($ND_{SedPipeEntrance}$)
- y: $sedtankinlepipe1_{origin1}$ - outerradius($ND_{SedPipeEntrance}$)
- z: $sedtankpipe4_{origin2}$)

layerset - $Layer_{set}$ selects the layer "0".

layerset <- $layer_{set}$("0")

layerfreezec - $Layer_{freeze}$ locks the layer "channel" so that it cannot be edited.

layerfreezec <- $layer_{freeze}$("channel")

layerthawis - $Layer_{thaw}$ unlocks the layer "inletslopes" so that it can be edited.

layerthawis <- $layer_{thaw}$("inletslopes")

subtractis - SubtractH subtracts one object from the other based on three points.

subtractis <-subtractH($Plant_{Origin}$,p1,p2)

$Plant_{Origin}$ =

Note: p1 and p2 are dummy variable used only in the program help section to designate the matrix below.

p1 =

- x: $sedtankinletpipe2_{origin0}$ + outerradius($ND_{SedPipeEntrance}$)
- y: $sedtankinletpipe2_{origin1}$ + outerradius($ND_{SedPipeEntrance}$)
- z: $sedtankpipe1_{origin2}$

p2 =

- x: $sedtankinletpipe1_{origin0}$ - outerradius($ND_{SedPipeEntrance}$)
- y:$sedtankinletpipe1_{origin1}$ - outerradius($ND_{SedPipeEntrance}$)
- z: $sedtankpipe4_{origin2}$

layerfreezeis - $Layer_{freeze}$ locks the layer "inletslopes" so that it cannot be edited.

layerfreezeis <- $layer_{freeze}$("inletslopes~)

layernewip - $Layer_{new}$ creates a new blue layer "inletpipes."

layernewip <- $layer_{new}$("inletpipes",blue)

slopesip - Calls the Sedimentation Tank Slopes Program to draw tank slopes.

slopesip <- sedslope($Plant_{Origin}$,p1,$AN_{SedTopInlet}$)

$Plant_{Origin}$ =

Note: p1 is a dummy variable used only in the program help section to designate the matrix below.

p1 =

- x: L_{Sed}
- y: W_{Sed}
- z: H_{Sed}

$An_{SedTopInlet}$ - Angle specifying the slope of the sedimentation tank walls.

moveip - Moveall is used to move all the selected objects by the specified distance.

moveip <-moveall($Plant_{Origin}$,p1)

Note: p1 is a dummy variable used only in the program help section to designate the matrix below.

p1 =

- x: $Plant_{Origin0}$

- y: $Plant_{Origin1}$
- z: $Plant_{Origin2}$ + outerdiameter($ND_{SedSludge}$)

pipe1 - Calls the [Pipe Program|AutoCad Pipe Program to draw a pipe.

pipe1 <- Pipe($sedtankinletpipe1_{origin}$,$ND_{SedPipeEntrance}$,-$sedtankinletpipe_{length}$,$EN_{PipeSpec}$)

$sedtankinletpipe1_{origin}$ =

- x: $Plant_{Origin0}$ - L_{Sed} + (.5$W_{InletChannel}$)
- y: $Plant_{Origin1}$ + (W_{Sed}/4)
- z: $Plant_{Origin2}$ + H_{Sed} - $H_{InletChannel}$

$ND_{SedPipeEntrance}$ = The nominal diameter of the tee. This value along with the pipe schedule is used to determine other actual dimensions of the tee.

$sedtankinletpipe_{length}$ = Specifies the length of the pipe.

$EN_{PipeSpec}$ = The enumerated pipe schedule type. Each schedule of pipe is represented by a specific number within our code.

pipe2 - Calls the [Pipe Program|AutoCad Pipe Program to draw a pipe.

pipe2 <- Pipe($sedtankinletpipe2_{origin}$,$ND_{SedPipeEntrance,-sedtankinletpipe}$length~,$EN_{PipeSpec}$)

$sedtankinletpipe2_{origin}$ =

- x: $Plant_{Origin0}$ - L_{Sed} + (.5$W_{InletChannel}$)
- y: $Plant_{Origin1}$ + (W_{Sed}/4)
- z: $Plant_{Origin2}$ + H_{Sed} - $H_{InletChannel}$

$ND_{SedPipeEntrance}$ = The nominal diameter of the tee. This value along with the pipe schedule is used to determine other actual dimensions of the tee.

$sedtankinletpipe_{length}$ = Specifies the length of the pipe.

$EN_{PipeSpec}$ = The enumerated pipe schedule type. Each schedule of pipe is represented by a specific number within our code.

subtractip -SubtractK subtracts one object from the other based on three points.

subtractip <- subtractK(p1,p2,$Plant_{Origin}$)

Note: p1 and p2 are dummy variable used only in the program help section to designate the matrix below.

p1 =

- x: $sedtankinletpipe1_{origin0}$ + outerradius(ND{~}SedPipeEntrance)
- y: $sedtankinletpipe1_{origin1}$
- z: $sedtankinletpipe1_{origin2}$

p2 =

- x: $sedtankinletpipe2_{origin0}$ + outerradius($ND_{SedPipeEntrance}$)
- y: $sedtankinletpipe2_{origin1}$
- z: $sedtankinletpipe2_{origin2}$

$Plant_{Origin}$ =

layerfreezeip - $Layer_{freeze}$ locks the layer "inletpipes" so that it cannot be edited.

layerfreezeip <- $layer_{freeze}$("inletpipes")

This page last changed on Nov 24, 2008 by ar329.

Inlet Channel Drawing Script

layer2 - $Layer_{new}$ creates a new light grey layer, "channel."

layer2 <- $layer_{new}$("channel",ltgrey)

channel1 - Calls the Channel Program to draw a channel.

channel1 <- ChannelDrawing($channel_{origin}$,p1,$T_{ChannelWall}$)

$channel_{origin}$ =

if layout 2:

- x: $Plant_{Origin0}$ - L_{Sed} + $W_{InletChannel}$
- y: $Plant_{Origin1}$ - $T_{PlantWall}$
- z: $Plant_{Origin2}$ + H_{Sed} - $H_{InletChannel}$

if layout 3:

- x: $Plant_{Origin0}$ - L_{Sed} + $W_{InletChannel}$
- y: $Plant_{Origin1}$ - $T_{PlantWall}$
- z: $Plant_{Origin2}$ + H_{Sed} - $H_{InletChannel}$

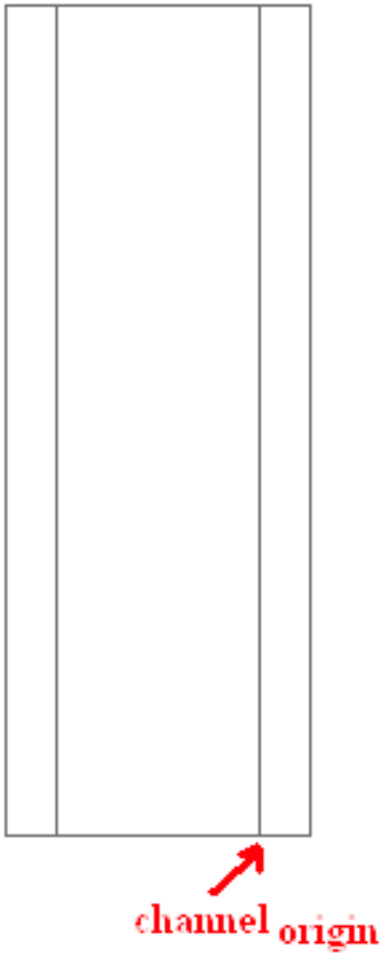
if layout 4 :

- x: $Plant_{Origin0}$ - L_{Sed} + $W_{ChannelInlet}$
- y: $Plant_{Origin1}$ - $T_{PlantWall}$
- z: $Plant_{Origin2}$ + H_{Sed} - $H_{InletChannel}$

p1 = Specifies the dimensions of the channel cutout.

- x: W_{Sed} - $2T_{PlantWall}$
- y: $W_{InletChannel}$
- z: $H_{InletChannel}$

$T_{Channel}$ specifies the thickness of the channel wall.



Top View

rotate1 - Rotate rotates the selected object by the designated degrees.

rotate1 <- rotate($channel_{origin}$,90)

$channel_{origin}$ =

if layout 2:

- x: $Plant_{Origin0}$ - L_{Sed} + $W_{InletChannel}$
- y: $Plant_{Origin1}$ - $T_{PlantWall}$
- z: $Plant_{Origin2}$ + H_{Sed} - $H_{InletChannel}$

if layout 3:

- x: $Plant_{Origin0}$ - L_{Sed} + $W_{InletChannel}$
- y: $Plant_{Origin1}$ - $T_{PlantWall}$
- z: $Plant_{Origin2}$ + H_{Sed} - $H_{InletChannel}$

if layout 4 :

- x: $Plant_{Origin0}$ - L_{Sed} + $W_{ChannelInlet}$
- y: $Plant_{Origin1}$ - $T_{PlantWall}$
- z: $Plant_{Origin2}$ + H_{Sed} - $H_{InletChannel}$

90 = rotation angle
box1 - Creates a box based on two points.

box1 <- box($channelinletwall_{origin}$,$channelinletwall_{origin}$ + $channelinletwall_{dim}$)

$channelinletwall_{origin}$ =

- x: $Plant_{Origin0}$ - L_{Sed} + $W_{ChannelInlet}$
- y: $Plant_{Origin1}$
- z: $Plant_{Origin2}$

$channelinletwall_{dim}$ =

- x: $T_{ChannelWall}$
- y: W_{Sed}
- z: H_{Sed}

bigunion - $Union_{allA}$ selects all the objects visible in the workspace and joins them all into a single object.

bigunion <- $union_{allA}$

box2 - Creates a box based on two points.

box2 <- box($channelinletbox1_{origin}$,$channelinletbox1_{origin}$ + $channelinletbox1_{dim}$)

$channelinletbox1_{origin}$ =

- x: $Plant_{Origin0}$ - L_{Sed} - $T_{PlantWall}$
- y: $Plant_{Origin1}$ + ($.5W_{Sed}$) - ($.5L_{ChannelInlet}$)
- z: $Plant_{Origin2}$ + H_{Sed} - $H_{InletChannel}$

$channelinletbox1_{dim}$ =

- x: $T_{PlantWall}$
- y: $L_{ChannelInlet}$
- z: $H_{InletChannel}$

subtract2 - SubtractD subtracts one object from the other based on two points.

subtract2 <- subtractD($Plant_{Origin}$,$channelinletbox1_{origin}$)

$Plant_{Origin}$ =

$channelinletbox1_{origin}$ =

- x: $Plant_{Origin0}$ - L_{Sed} - $T_{PlantWall}$
- y: $Plant_{Origin1}$ + ($.5W_{Sed}$) - ($.5L_{ChannelInlet}$)
- z: $Plant_{Origin2}$ + H_{Sed} - $H_{InletChannel}$

box3 - Creates a box based on two points.

box3 <- box($channelinletbox2_{origin}$,$channelinletbox2_{origin}$ + $channelinletbox2_{dim}$)

$channelinletbox2_{origin}$ =

- x: $Plant_{Origin0}$ - L_{Sed}
- y: $Plant_{Origin1}$
- z: $Plant_{Origin2}$

channelinletbox2$_{dim}$ =

- x: $W_{ChannelInlet}$
- y: W_{Sed}
- z: H_{Sed}

subtract3 - SubtractD subtracts one object from the other based on two points.

subtract3 <- subtractD($Plant_{Origin}$,channelinletbox2$_{origin}$)

$Plant_{Origin}$ =

channelinletbox2$_{origin}$ =

- x: $Plant_{Origin0}$ - L_{Sed}
- y: $Plant_{Origin1}$
- z: $Plant_{Origin2}$

layerset - $Layer_{set}$ selects the layer "0".

layerset <- layer$_{set}$("0")

viewtops - Rotates the object so that it is viewed from the top.

viewtops <-viewtop1

layerfreeze2 - $Layer_{freeze}$ locks the layer "channel" so that it cannot be edited.

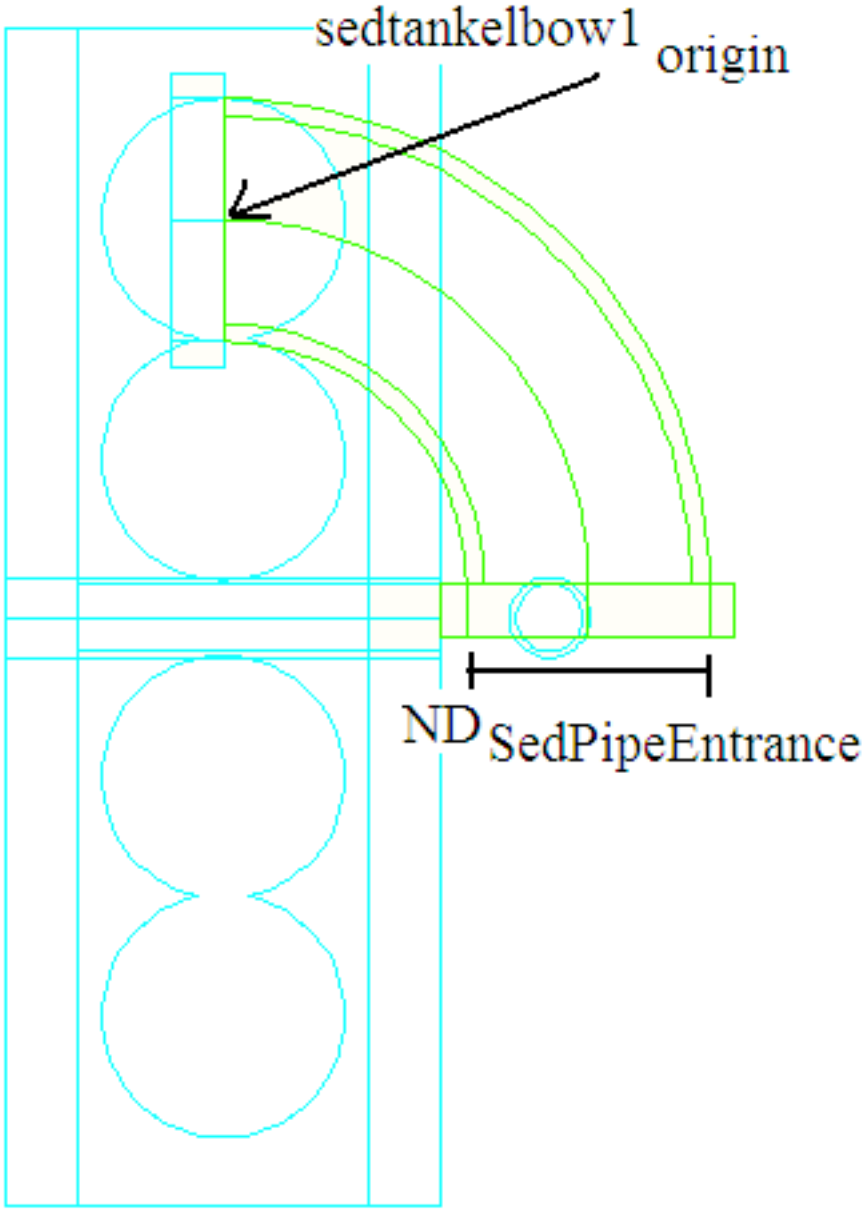
layerfreeze2 <- layer$_{freeze}$("channel")

This page last changed on Dec 16, 2008 by ar329.

Elbow Drawing Script

layer4 - Layer_{new} creates a new green layer "elbow."

layer4 <- layer_{new}("elbow",green)



Top View

elbow1 - Calls the Elbow Program to create an elbow.

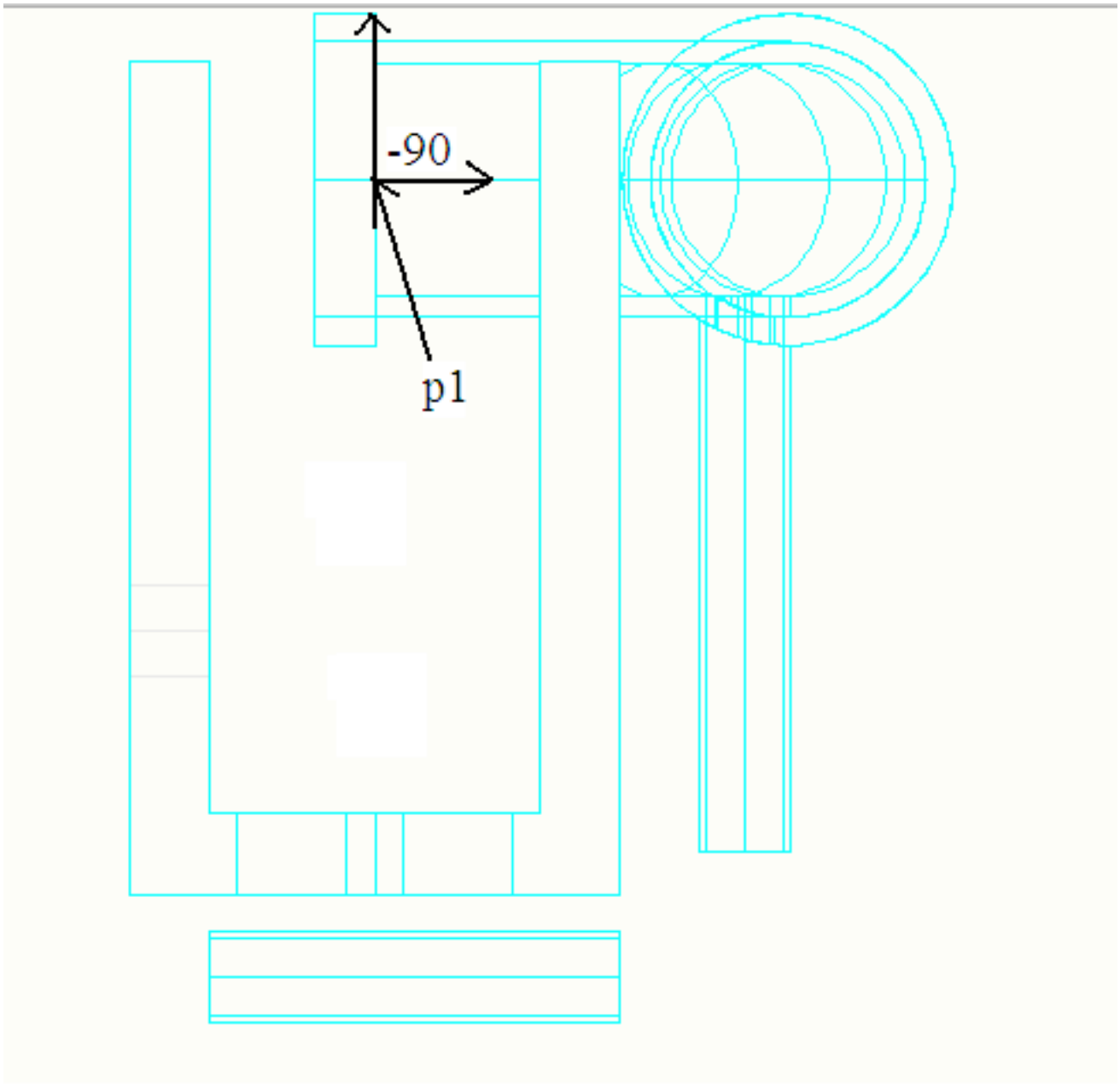
elbow1 <- elbow($\text{sedtankelbow1}_{origin}$,$\text{ND}_{SedPipeEntrance}$,$\text{EN}_{PipeSpec}$)

$\text{sedtankelbow1}_{origin}$ =

- x: $Plant_{Origin0}$ - L_{Sed} + $W_{InletChannel}/2$
- y: $Plant_{Origin1}$ + $W_{Sed}/2$ + outerradius($ND_{SedLaunder}$) + 3*($W_{Sed}/2$ - outerradius($ND_{SedLaunder}$))/4 - ElbowRadius($ND_{SedPipeEntrance}$)
- z: $Plant_{Origin2}$ + $sedtankpipe9_{length}$ + $S_{SedInlet}$ + 2ElbowRadius($ND_{SedPipeEntrance}$)

$ND_{SedPipeEntrance}$ = The nominal diameter of the pipe. This value along with the pipe schedule is used to determine other actual dimensions of the elbow

$EN_{PipeSpec}$ = The enumerated pipe schedule type. Each schedule of pipe is represented by a specific number within our code.



Top View

rotate6 - [$Rotate_{3d}$] turns the object based on a given axis and degree angle.

rotate6 <- $rotate_{3d}$(p1,p2,"x",-90)

Note: p1 and p2 are dummy variables used only in the program help section to designate the matrix below.

p1 =

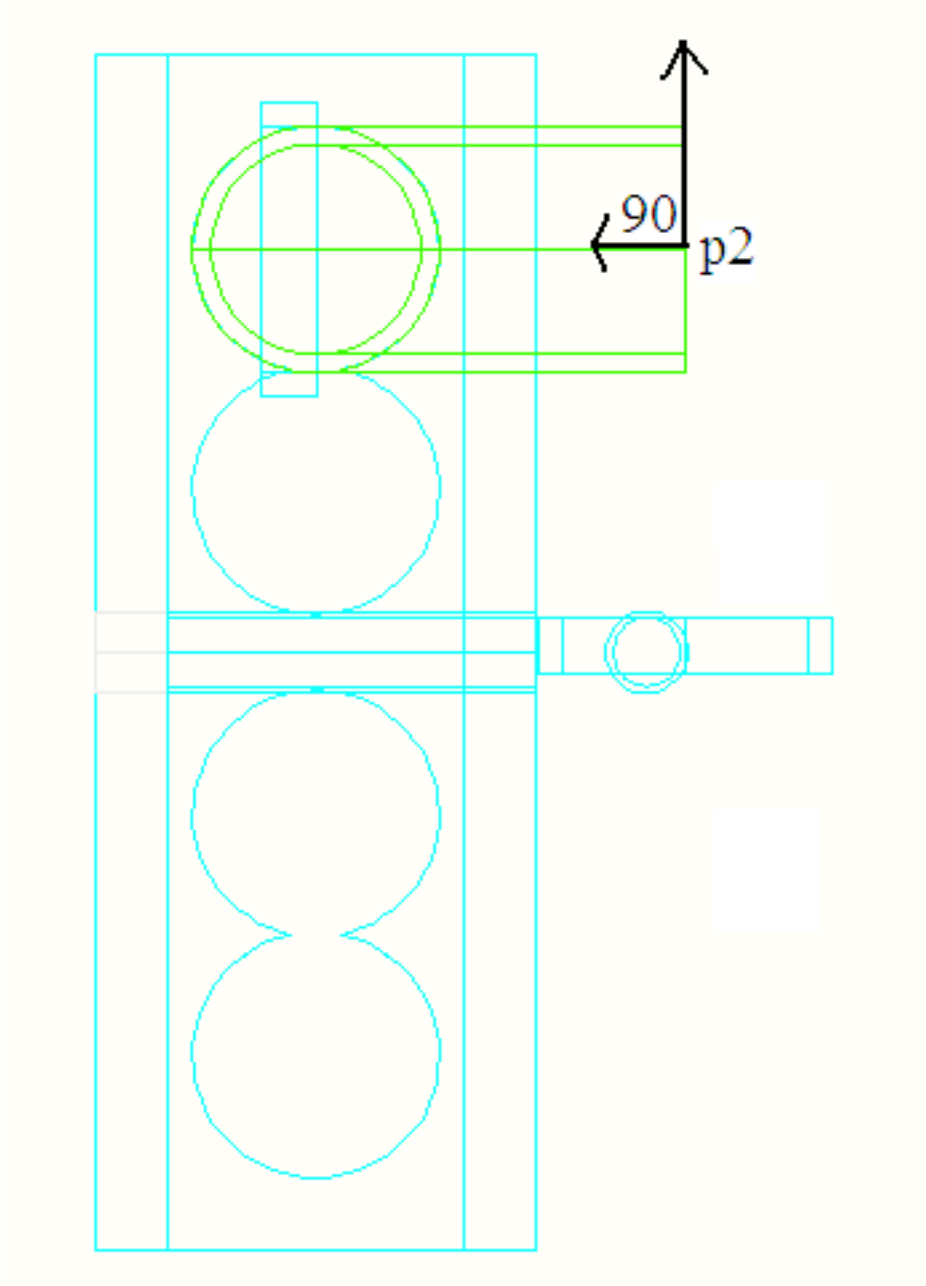
- x: $sedtankelbow1_{origin0}$
- y: $sedtankelbow1_{origin1}$ + ElbowRadius($ND_{SedPipeEntrance}$)
- z: $sedtankelbow1_{origin2}$ + innerD($ND_{SedPipeEntrance}$,$EN_{PipeSpec}$)/2

p2 =

- x: $sedtankelbow1_{origin0}$
- y: $sedtankelbow1_{origin1}$ + ElbowRadius($ND_{SedPipeEntrance}$)
- z: $sedtankelbow_{origin2}$

"x" - specifies axis that object will be rotated about.

-90 - rotation angle



Top View

rotate7 - Rotate$_{3d}$ turns the object based on a given axis and degree angle.

rotate7 <- rotate$_{3d}$(p1,p2,"y",90)

Note: p1 and p2 are dummy variables used only in the program help section to designate the matrix below.

p1 =

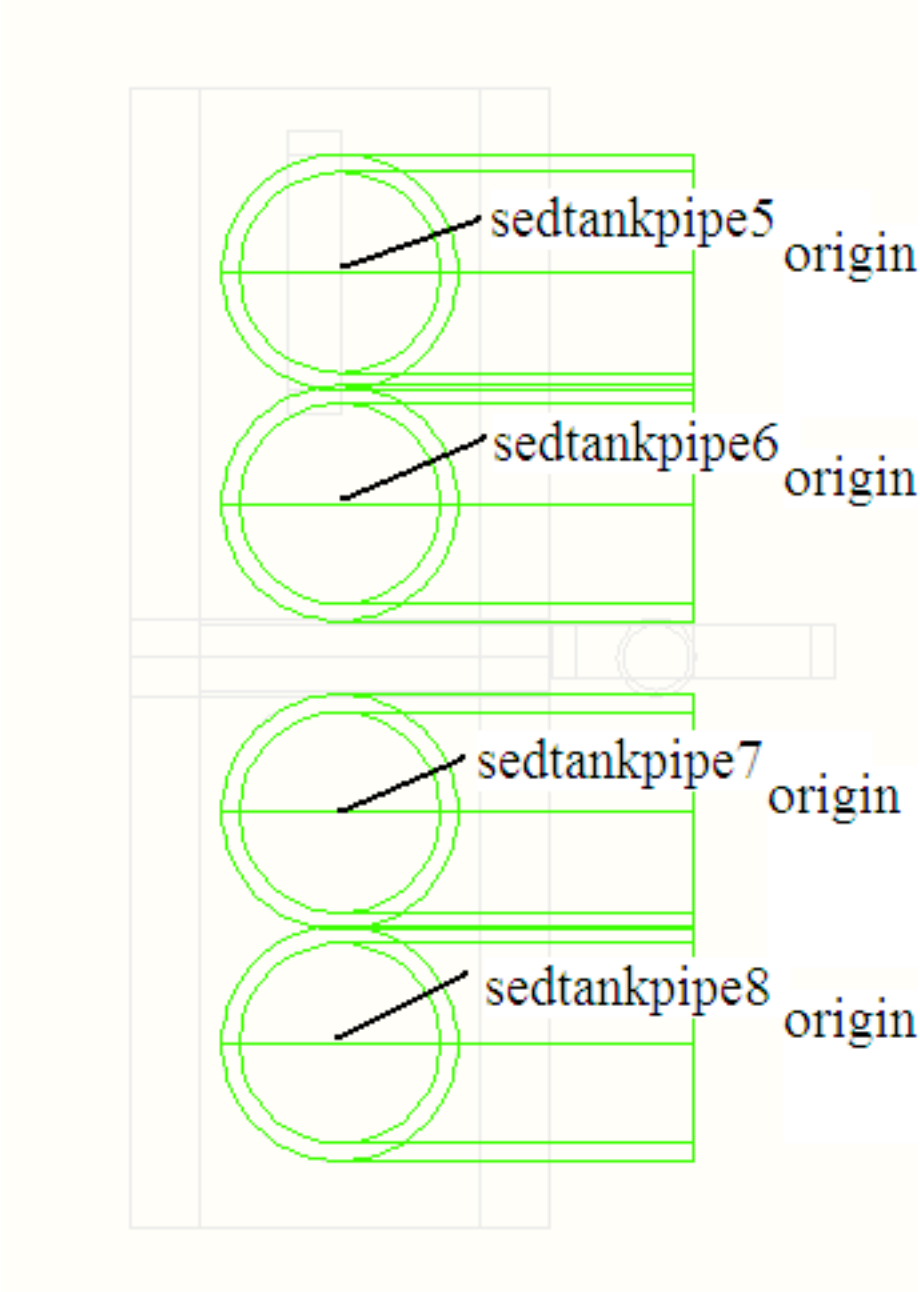
- x: sedtankelbow1$_{origin0}$
- y: sedtankelbow1$_{origin1}$ + ElbowRadius(ND$_{SedPipeEntrance}$)
- z: sedtankelbow1$_{origin2}$ + innerD(ND$_{SedPipeEntrance}$,EN$_{PipeSpec}$)/2

p2 =

- x: sedtankelbow1$_{origin0}$
- y: sedtankelbow1$_{origin1}$ + ElbowRadius(ND$_{SedPipeEntrance}$)
- z: sedtankelbow$_{origin2}$

"y" - specifies axis that object will be rotated about.

90 - rotation angle.



Top View

copy1 - CopyDa duplicates the selected object.

copy1 <- copyDa(p1,$sedtankpipe5_{origin}$,$sedtankpipe6_{origin}$,$sedtankpipe7_{origin}$,$sedtankpipe8_{origin}$)

Note: p1 is a dummy variable used only in the program help section to designate the matrix below.

p1 =

- x: $sedtankpipe5_{origin0}$
- y: $sedtankpipe5_{origin1}$
- z: $sedtankpipe5_{origin2}$ - outerradius($ND_{SedPipeEntrance}$)

$sedtankpipe5_{origin}$ =

- x: $Plant_{Origin0}$ - L_{Sed} + ElbowRadius($ND_{SedPipeEntrance}$)
- y: $Plant_{Origin1}$ + W_{Sed}/2 + outerradius($ND_{SedLaunder}$) + 3(W_{Sed}/2 - outerradius(ND{~}SedLaunder))/4

- z: $Plant_{Origin2}$ + (W_{Sed}/2)*tank($AN_{SedBottom}$) + ElbowRadius($ND_{SedPipeEntrance}$)

$sedtankpipe6_{origin}$ =

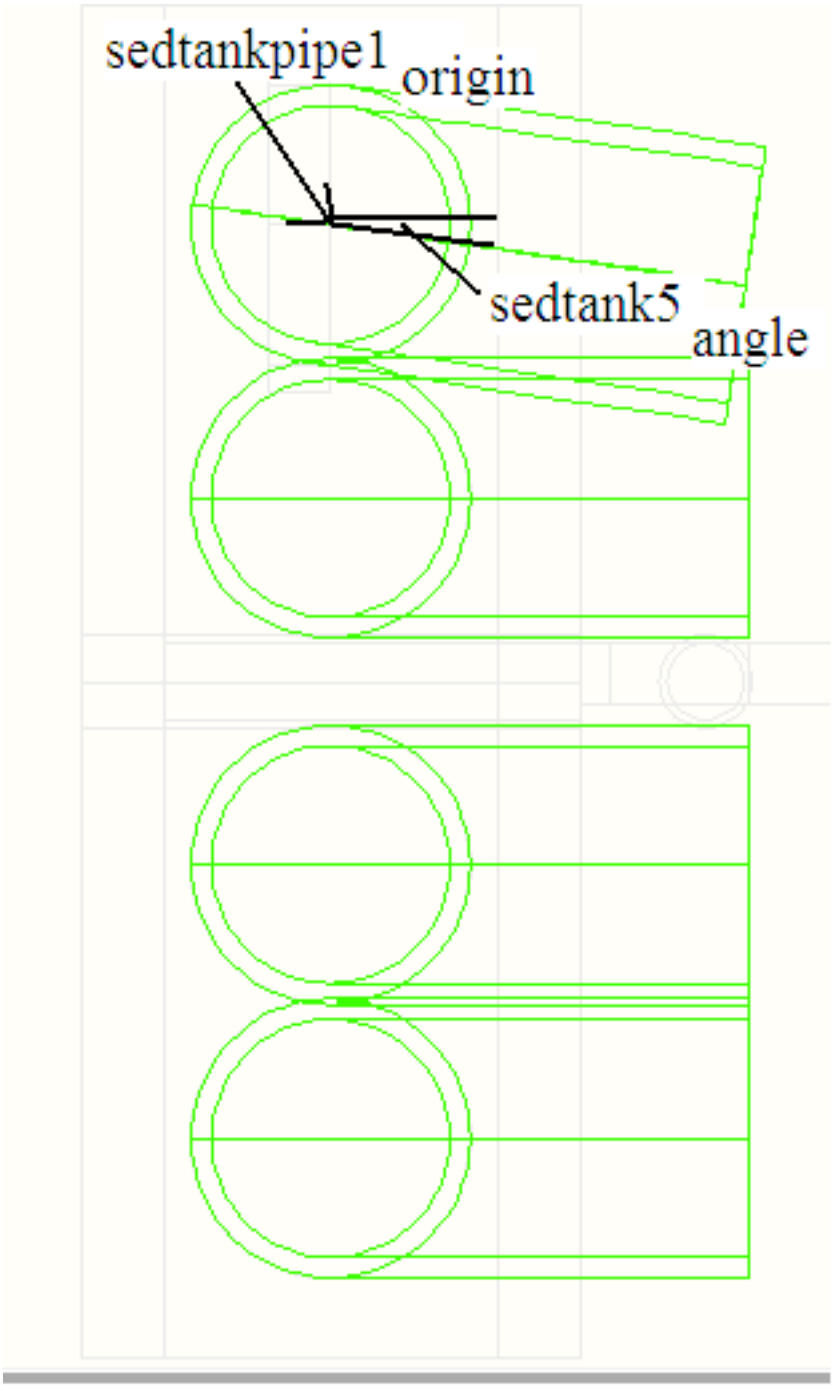
- x: $Plant_{Origin0}$ - L_{Sed} + ElbowRadius($ND_{SedPipeEntrance}$)
- y: $Plant_{Origin1}$ + W_{Sed}/2 + outerradius($ND_{SedLaunder}$) + 1(W_{Sed}/2 - outerradius(ND{~}SedLaunder))/4
- z: $Plant_{Origin2}$ + (W_{Sed}/2)*tank($AN_{SedBottom}$) + ElbowRadius($ND_{SedPipeEntrance}$)

$sedtankpipe7_{origin}$ =

- x: $Plant_{Origin0}$ - L_{Sed} + ElbowRadius($ND_{SedPipeEntrance}$)
- y: $Plant_{Origin1}$ + W_{Sed}/2 + outerradius($ND_{SedLaunder}$) + 1(W_{Sed}/2 - outerradius(ND{~}SedLaunder))/4
- z: $Plant_{Origin2}$ + (W_{Sed}/2)*tank($AN_{SedBottom}$) + ElbowRadius($ND_{SedPipeEntrance}$)

$sedtankpipe8_{origin}$ =

- x: $Plant_{Origin0}$ - L_{Sed} + ElbowRadius($ND_{SedPipeEntrance}$)
- y: $Plant_{Origin1}$ + W_{Sed}/2 + outerradius($ND_{SedLaunder}$) + 3(W_{Sed}/2 - outerradius(ND{~}SedLaunder))/4
- z: $Plant_{Origin2}$ + (W_{Sed}/2)*tank($AN_{SedBottom}$) + ElbowRadius($ND_{SedPipeEntrance}$)



Top View

rotate8 - Rotate$_{3d}$ turns the object based on a given axis and degree angle.

rotate8 <- rotate$_{3d}$(p1,sedtankpipe1$_{origin}$,"z",sedtankpipe5$_{angle}$)

Note: p1 is a dummy variable used only in the program help section to designate the matrix below.

p1 =

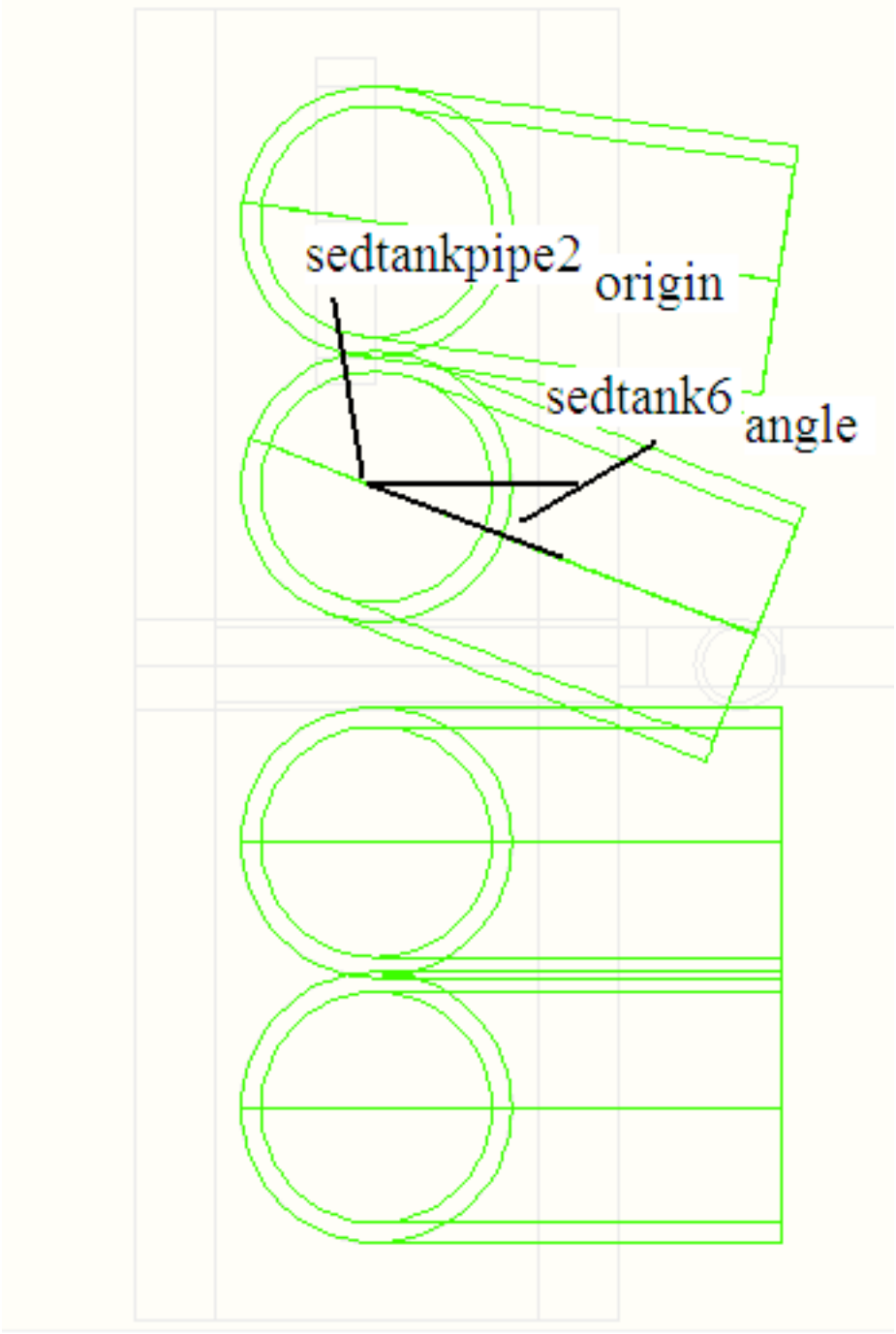
- x: sedtankpipe5$_{origin0}$
- y: sedtankpipe5$_{origin1}$
- z: sedtankpipe5$_{origin2}$ - outerradius(ND$_{SedPipeEntrance}$)

sedtankpipe1$_{origin}$ =

- x: $Plant_{Origin0}$ - L_{Sed} + ElbowRadius($ND_{SedPipeEntrance}$)
- y: $Plant_{Origin1}$ + W_{Sed}/2 + outerradius($ND_{SedLaunder}$) + 3(W_{Sed}/2 - outerradius(ND{~}SedLaunder))/4
- z: $Plant_{Origin2}$ + (W_{Sed}/2)*tank($AN_{SedBottom}$) + ElbowRadius($ND_{SedPipeEntrance}$)

"z" - specifies axis that object will be rotated about.

$sedtankpipe5_{angle}$ = -(rad/deg) * atan((((5*L_{Sed})/8 - $W_{Channel}$/2)/ [3 * (W_{Sed}/2 - outerradius($ND_{SedLaunder}$))/4) + outerradius($ND_{SedLaunder}$))$^{-1}$)



Top View

rotate9 - [$Rotate_{3d}$] turns the object based on a given axis and degree angle.

rotate9 <- $rotate_{3d}$(p1,$sedtankpipe2_{origin}$,"z".$sedtankpipe6_{angle}$)

Note: p1 is a dummy variable used only in the program help section to designate the matrix below.

p1 =

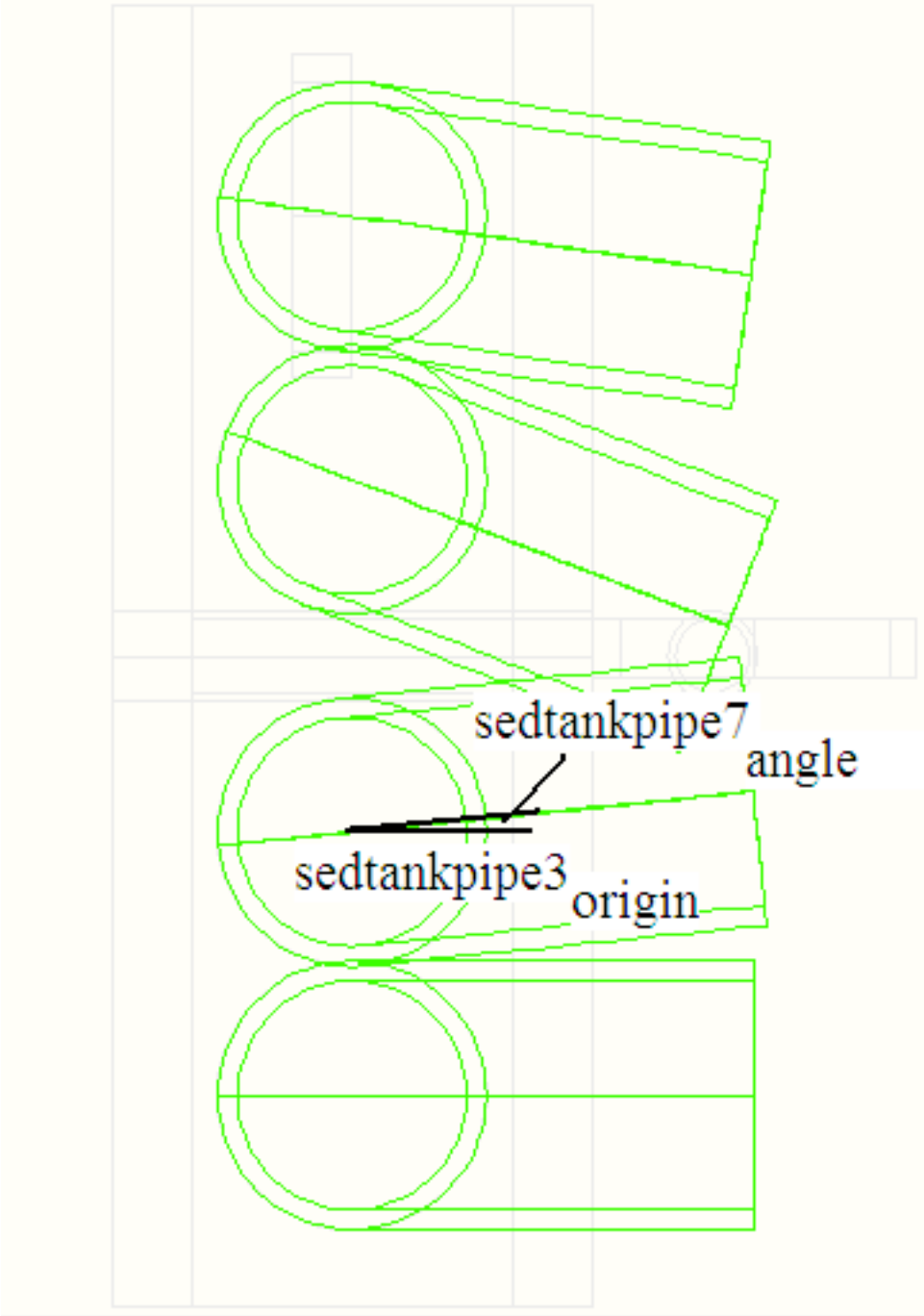
- x: $sedtankpipe6_{origin0}$
- y: $sedtankpipe6_{origin1}$
- z: $sedtankpipe6_{origin2}$ - outerradius($ND_{SedPipeEntrance}$)

$sedtankpipe2_{origin}$ =

- x: $Plant_{Origin0}$ - L_{Sed} + ElbowRadius($ND_{SedPipeEntrance}$)
- y: $Plant_{Origin1}$ + W_{Sed}/2 + outerradius($ND_{SedLaunder}$) + 1(W_{Sed}/2 - outerradius(ND{~}SedLaunder))/4
- z: $Plant_{Origin2}$ + (W_{Sed}/2)*tank($AN_{SedBottom}$) + ElbowRadius($ND_{SedPipeEntrance}$)

"z" - specifies axis that object will be rotated about.

$sedtankpipe6_{angle}$ = -(rad/deg) * ((((L_{Sed}/8 - $W_{Channel}$/2)/ ((W_{Sed}/2 - outerradius($ND_{SedLaunder}$))/4) + outerradius($ND_{SedLaunder}$))$^{-1}$)



Top View

rotate10 - Rotate$_{3d}$ turns the object based on a given axis and degree angle.

rotate10 <- rotate$_{3d}$(p1,sedtankpipe3$_{origin}$,"z".sedtankpipe7$_{angle}$)

Note: p1 is a dummy variable used only in the program help section to designate the matrix below.

p1 =

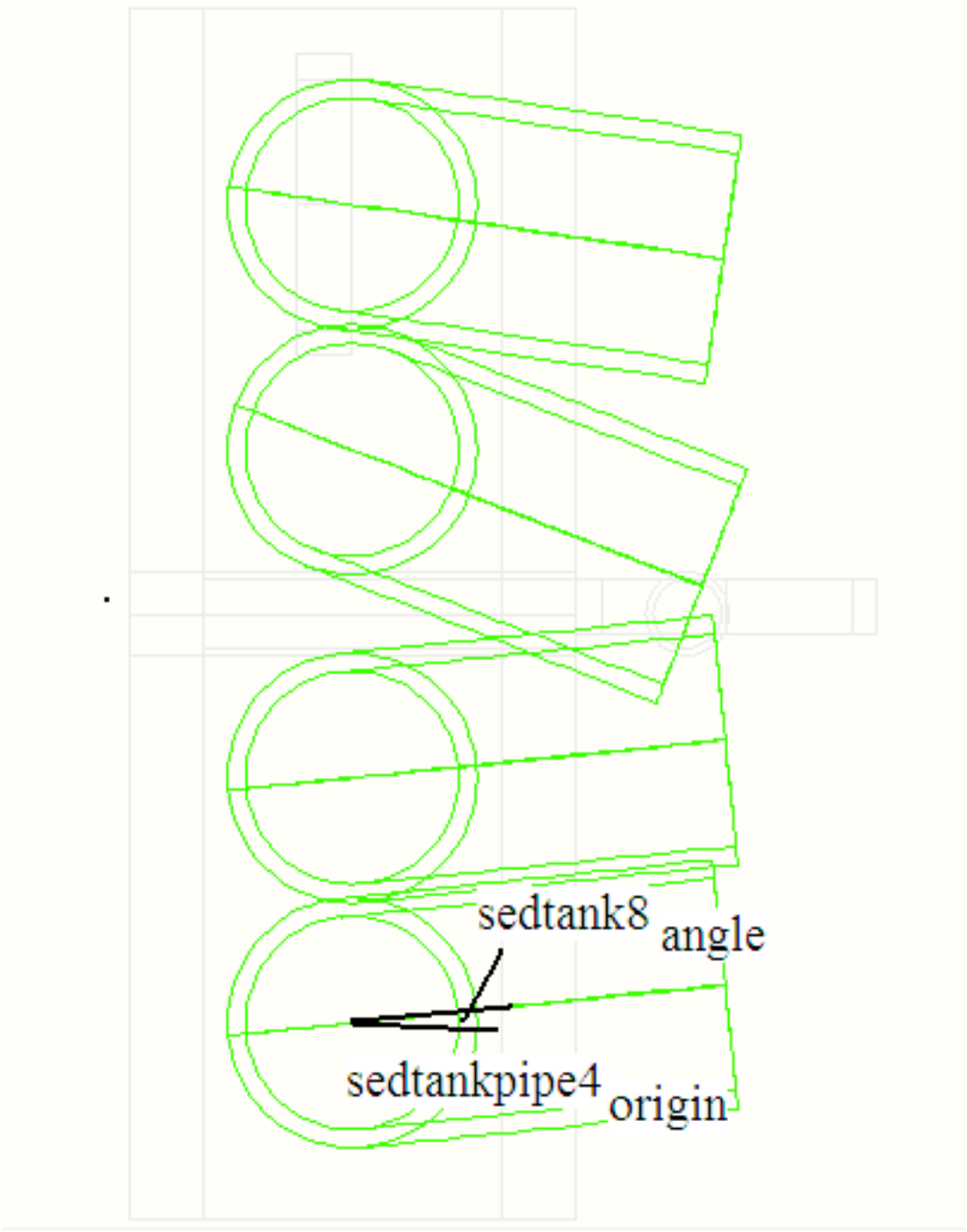
- x: sedtankpipe7$_{origin0}$
- y: sedtankpipe7$_{origin1}$
- z: sedtankpipe7$_{origin2}$ - outerradius(ND$_{SedPipeEntrance}$)

sedtankpipe3$_{origin}$ =

- x: $Plant_{Origin0}$ - L_{Sed} + ElbowRadius($ND_{SedPipeEntrance}$)
- y: $Plant_{Origin1}$ + W_{Sed}/2 + outerradius($ND_{SedLaunder}$) + 1(W_{Sed}/2 - outerradius(ND{~}SedLaunder))/4
- z: $Plant_{Origin2}$ + (W_{Sed}/2)*tank($AN_{SedBottom}$) + ElbowRadius($ND_{SedPipeEntrance}$)

"z" - specifies axis that object will be rotated about.

$sedtankpipe7_{angle}$ = -(rad/deg) * atan(((3*L_{Sed}/8 - $W_{Channel}$/2)/ ((W_{Sed}/2 - outerradius($ND_{SedLaunder}$))/4) + outerradius($ND_{SedLaunder}$))$^{-1}$)



Top View

rotate11 - [$Rotate_{3d}$] turns the object based on a given axis and degree angle.

rotate11 <- $rotate_{3d}$(p1,$sedtankpipe4_{origin}$,"z".$sedtankpipe8_{angle}$)

Note: p1 is a dummy variable used only in the program help section to designate the matrix below.

p1 =

- x: $sedtankpipe8_{origin0}$
- y: $sedtankpipe8_{origin1}$
- z: $sedtankpipe8_{origin2}$ - outerradius($ND_{SedPipeEntrance}$)

$sedtankpipe4_{origin}$ =

- x: $Plant_{Origin0}$ - L_{Sed} + ElbowRadius($ND_{SedPipeEntrance}$)
- y: $Plant_{Origin1}$ + W_{Sed}/2 + outerradius($ND_{SedLaunder}$) + 1(W_{Sed}/2 - outerradius(ND{~}SedLaunder))/4
- z: $Plant_{Origin2}$ + (W_{Sed}/2)*tank($AN_{SedBottom}$) + ElbowRadius($ND_{SedPipeEntrance}$)

"z" - specifies axis that object will be rotated about.

$sedtankpipe8_{angle}$ = -(rad/deg) * atan((((7*L_{Sed})/8 - $W_{Channel}$/2)/ (3*(W_{Sed}/2 - outerradius($ND_{SedLaunder}$))/4) + outerradius($ND_{SedLaunder}$))$^{-1}$)
layerset - $Layer_{set}$ selects the layer "0".

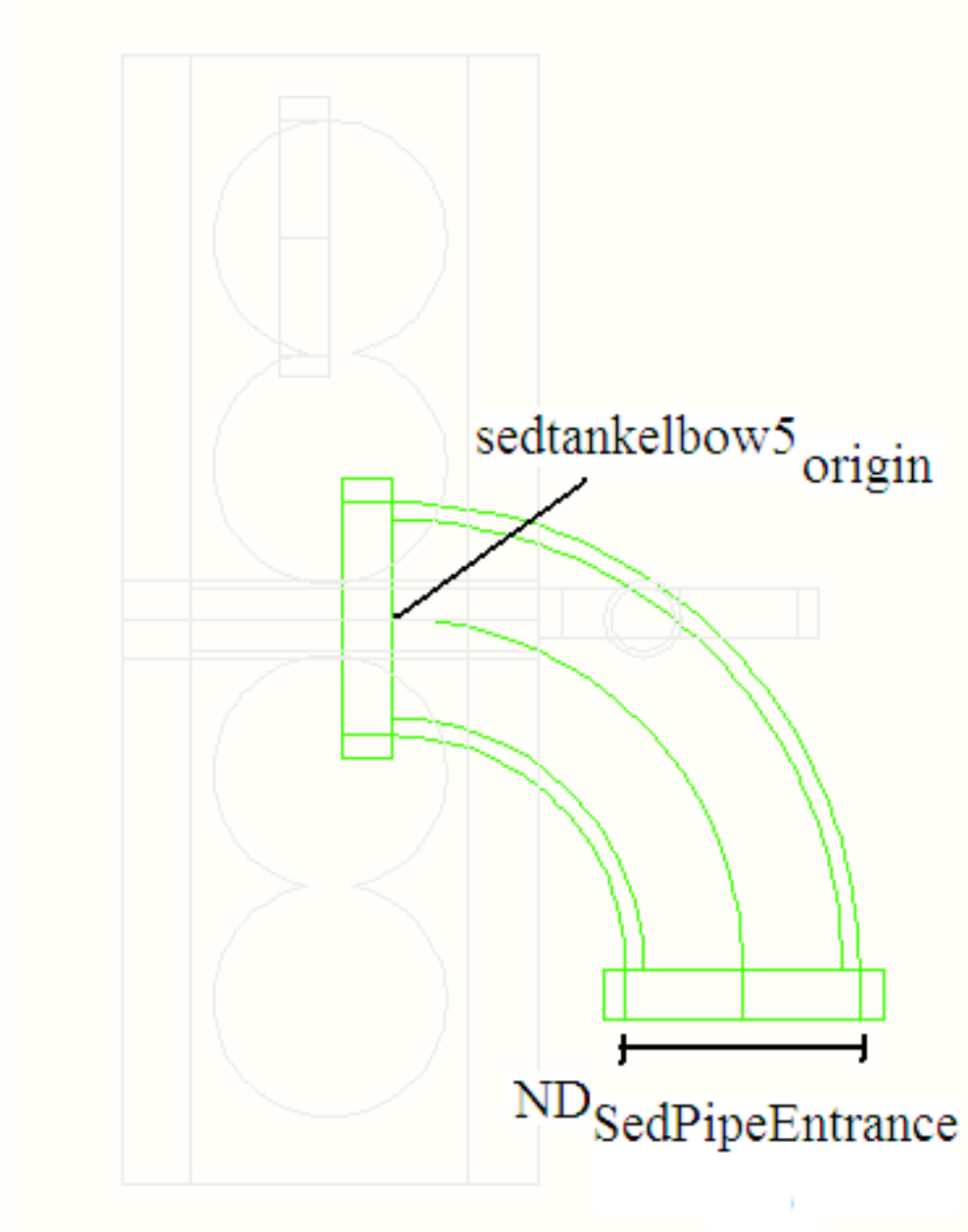
layerset <- $layer_{set}$("0")

layerfreeze4 - $Layer_{freeze}$ locks the layer "elbow" so that it cannot be edited.

layerfreeze4 <- $layer_{freeze}$("elbow")

layer5 - $Layer_{new}$ creates a new green layer "elbowa."

layer5 <- $layer_{new}$("elbowa",green)



Top View

elbow5 - Calls the Elbow Program to create an elbow.

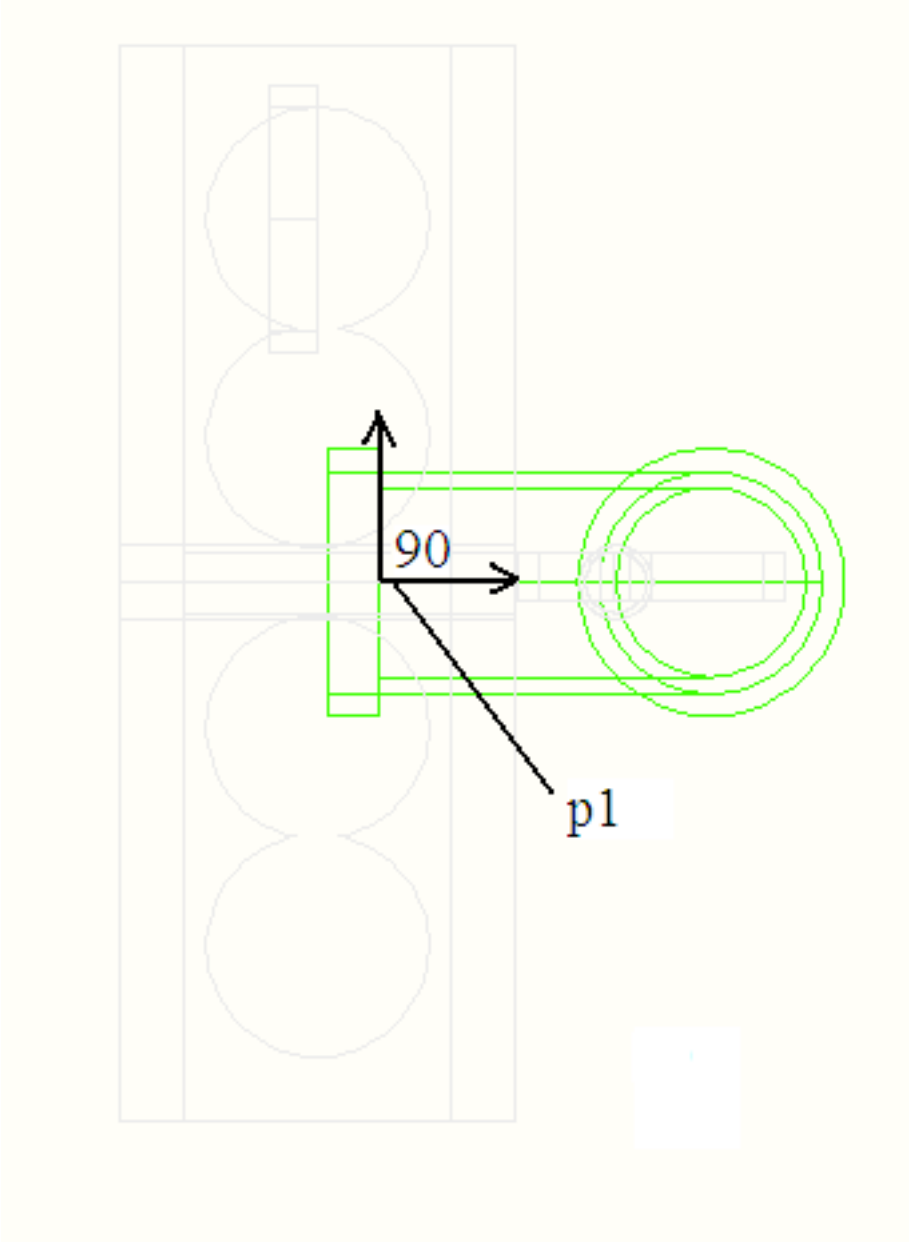
elbow5 <- elbow($sedtankelbow5_{origin}$,$ND_{SedPipeEntrance}$,$EN_{PipeSpec}$)

$sedtankelbow5_{origin}$ =

- x: $tank_{origin0}$ - (7*L_{Sed})/8 - ElbowRadius($ND_{SedPipeEntrance}$)
- y: $tank_{origin1}$ + W_{Sed}/2 - ElbowRadius($ND_{SedPipeEntrance}$)
- z: $tank_{origin2}$ + $S_{SedInlet}$ + $sedtankpipe9_{length}$ + ElbowRadius($ND_{SedPipeEntrance}$)

$ND_{SedPipeEntrance}$ = The nominal diameter of the pipe. This value along with the pipe schedule is used to determine other actual dimensions of the elbow

$EN_{PipeSpec}$ = The enumerated pipe schedule type. Each schedule of pipe is represented by a specific number within our code.



Top View

rotate14 - Rotate$_{3d}$ turns the object based on a given axis and degree angle.

rotate14 <- rotate$_{3d}$(p1,p2,"x",90)

Note: p1 and p2 are dummy variables used only in the program help section to designate the matrix below.

p1 =

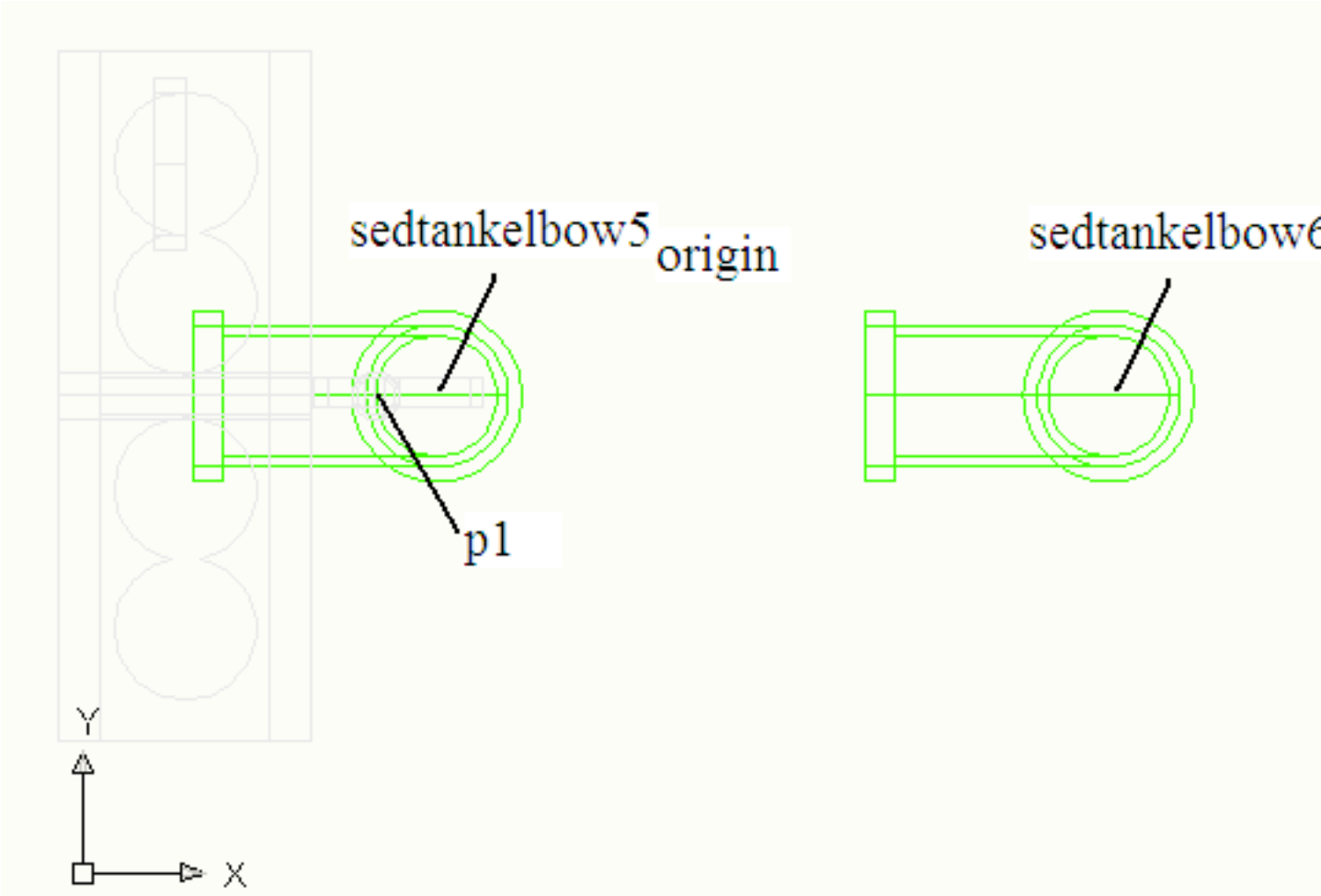
- x: sedtankelbow5$_{origin0}$
- y: sedtankelbow5$_{origin1}$ + ElbowRadius(ND$_{SedPipeEntrance}$)
- z: sedtankelbow5$_{origin2}$ + (innerD(ND$_{SedPipeEntrance}$,EN$_{PipeSpec}$)/2)

p2 =

- x: sedtankelbow5$_{origin0}$
- y: sedtankelbow5$_{origin1}$ + ElbowRadius(ND$_{SedPipeEntrance}$)
- z: sedtankelbow5$_{origin2}$

"x" - specifies axis that object will be rotated about.

90 - specifies rotation angle.



Top View

copy2 - CopyDa duplicates the selected objuect.

copy2 <- copyDa(p1,sedtankelbow5$_{origin}$,sedtankelbow6$_{origin}$,sedtankelbow7$_{origin}$,sedtankelbow8$_{origin}$)

Note: p1 is a dummy variable used only in the program help section to designate the matrix below.

p1 =

- x: sedtankpipe9$_{origin0}$ - ElbowRadius(ND$_{SedPipeEntrance}$)
- y: sedtankpipe9$_{origin1}$
- z: sedtankpipe9$_{origin2}$

sedtankelbow5$_{origin}$ =

- x: tank$_{origin0}$ - (7*L$_{Sed}$)/8 - ElbowRadius(ND$_{SedPipeEntrance}$)
- y: tank$_{origin1}$ + W$_{Sed}$/2 - ElbowRadius(ND$_{SedPipeEntrance}$)
- z: tank$_{origin2}$ + S$_{SedInlet}$ + sedtankpipe9$_{length}$ + ElbowRadius(ND$_{SedPipeEntrance}$)

sedtankelbow6$_{origin}$ =

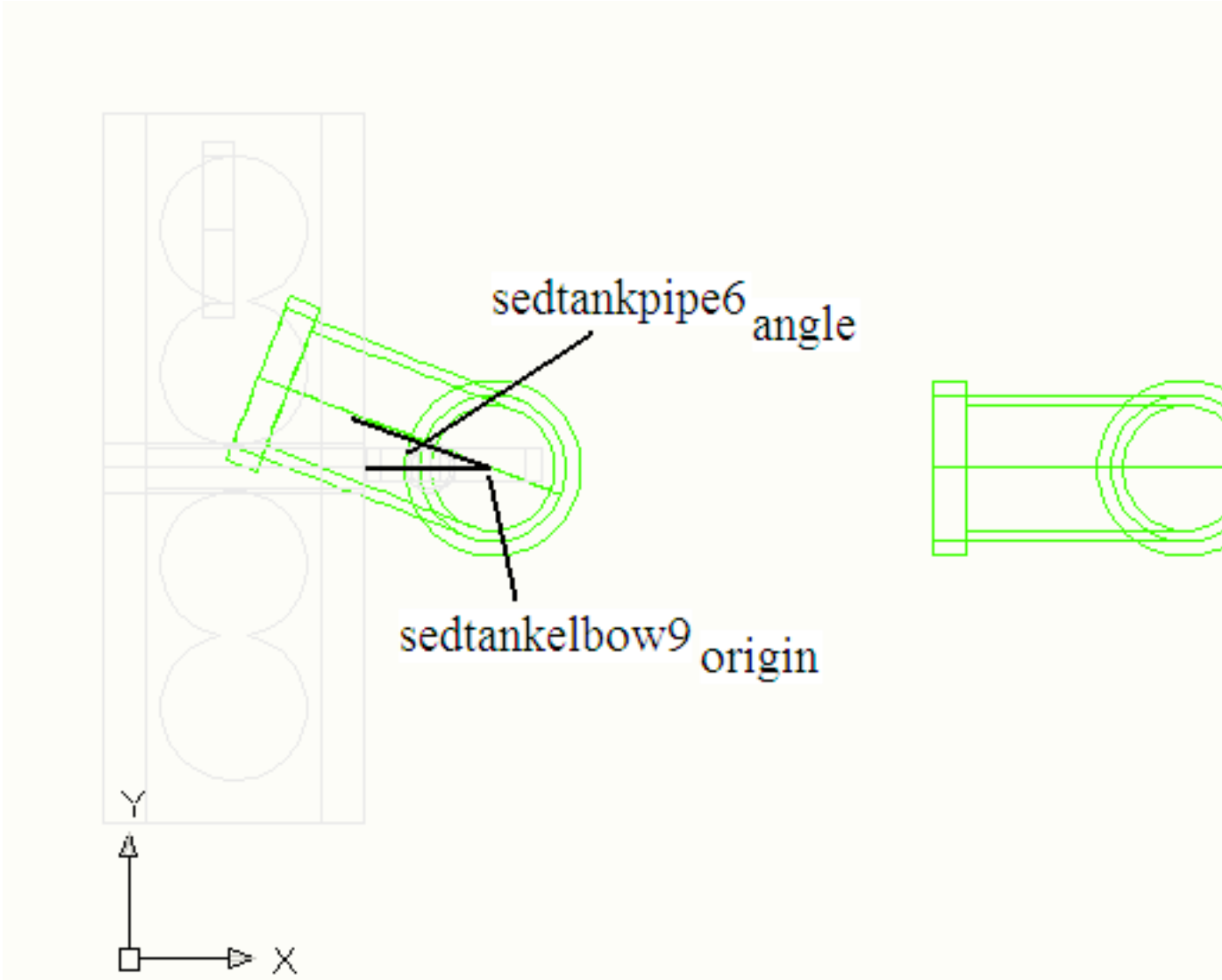
- x: tank$_{origin0}$ - (5*L$_{Sed}$)/8 - ElbowRadius(ND$_{SedPipeEntrance}$)
- y: tank$_{origin1}$ + W$_{Sed}$/2 - ElbowRadius(ND$_{SedPipeEntrance}$)
- z: tank$_{origin2}$ + S$_{SedInlet}$ + sedtankpipe9$_{length}$ + ElbowRadius(ND$_{SedPipeEntrance}$)

$sedtankelbow7_{origin}$ =

- x: $tank_{origin0}$ - (3*L_{Sed})/8 - ElbowRadius($ND_{SedPipeEntrance}$)
- y: $tank_{origin1}$ + W_{Sed}/2 - ElbowRadius($ND_{SedPipeEntrance}$)
- z: $tank_{origin2}$ + $S_{SedInlet}$ + $sedtankpipe9_{length}$ + ElbowRadius($ND_{SedPipeEntrance}$)

$sedtankelbow8_{origin}$ =

- x: $tank_{origin0}$ - (1*L_{Sed})/8 - ElbowRadius($ND_{SedPipeEntrance}$)
- y: $tank_{origin1}$ + W_{Sed}/2 - ElbowRadius($ND_{SedPipeEntrance}$)
- z: $tank_{origin2}$ + $S_{SedInlet}$ + $sedtankpipe9_{length}$ + ElbowRadius($ND_{SedPipeEntrance}$



Top View

rotate15 - [$Rotate_{3d}$] turns the object based on a given axis and degree angle.

rotate15 <- $rotate_{3d}$(p1,$sedtankpipe9_{origin}$,"z".$sedtankpipe6_{angle}$)

Note: p1 is a dummy variable used only in the program help section to designate the matrix below.

p1 =

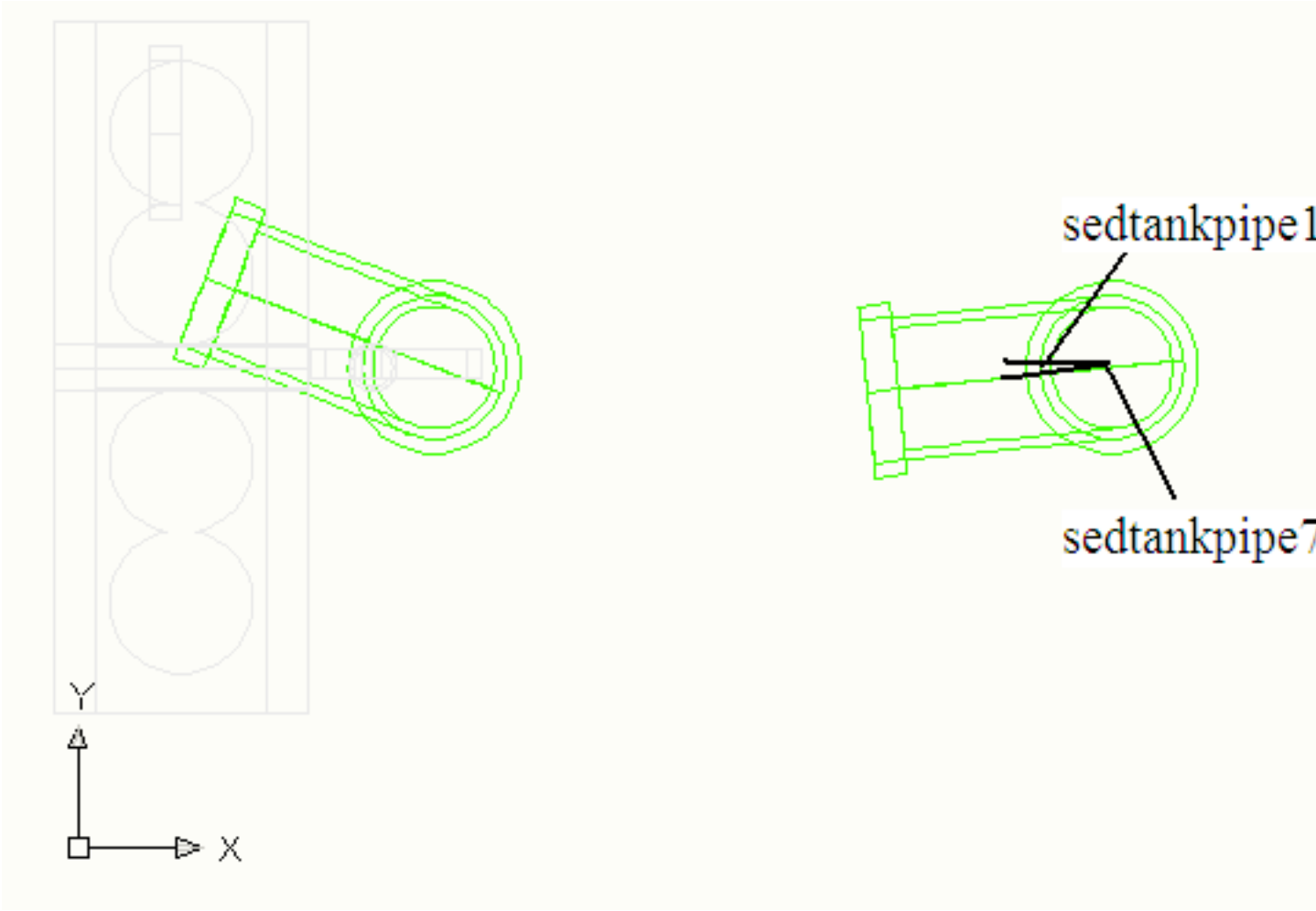
- x: $sedtankpipe9_{origin1}$ - outerradius($ND_{SedPipeEntrance}$)
- y: $sedtankpipe9_{origin2}$
- z: $sedtankpipe10_{origin0}$

$sedtankelbow9_{origin}$ =

- x: $tank_{origin0}$ - L_{Sed} + $W_{Channel}$ + $T_{ChannelWall}$ + ElbowRadius(ND~SedLaunder~)
- y: $tank_{origin1}$ + $W_{Sed}/2$ - ElbowRadius($ND_{SedLaunder}$)
- z: $tank_{origin2}$ + HW_{Sed} - $H_{SedAbove}/2$ - ElbowRadius($ND_{SedLaunder}$)

"z" - specifies axis that object will be rotated about.

$sedtankpipe6_{angle}$ = -(rad/deg) * (((($L_{Sed}/8$ - $W_{Channel}/2$)/ (($W_{Sed}/2$ - outerradius($ND_{SedLaunder}$))/4) + outerradius($ND_{SedLaunder}$))$^{-1}$)



Top View

rotate16 - [$Rotate_{3d}$] turns the object based on a given axis and degree angle.

rotate16 <- $rotate_{3d}$(p1,$sedtankpipe10_{origin}$,"z".$sedtankpipe7_{angle}$)

Note: p1 is a dummy variable used only in the program help section to designate the matrix below.

p1 =

- x: $sedtankpipe10_{origin1}$ - outerradius($ND_{SedPipeEntrance}$)
- y: $sedtankpipe10_{origin2}$
- z: $sedtankpipe11_{origin0}$

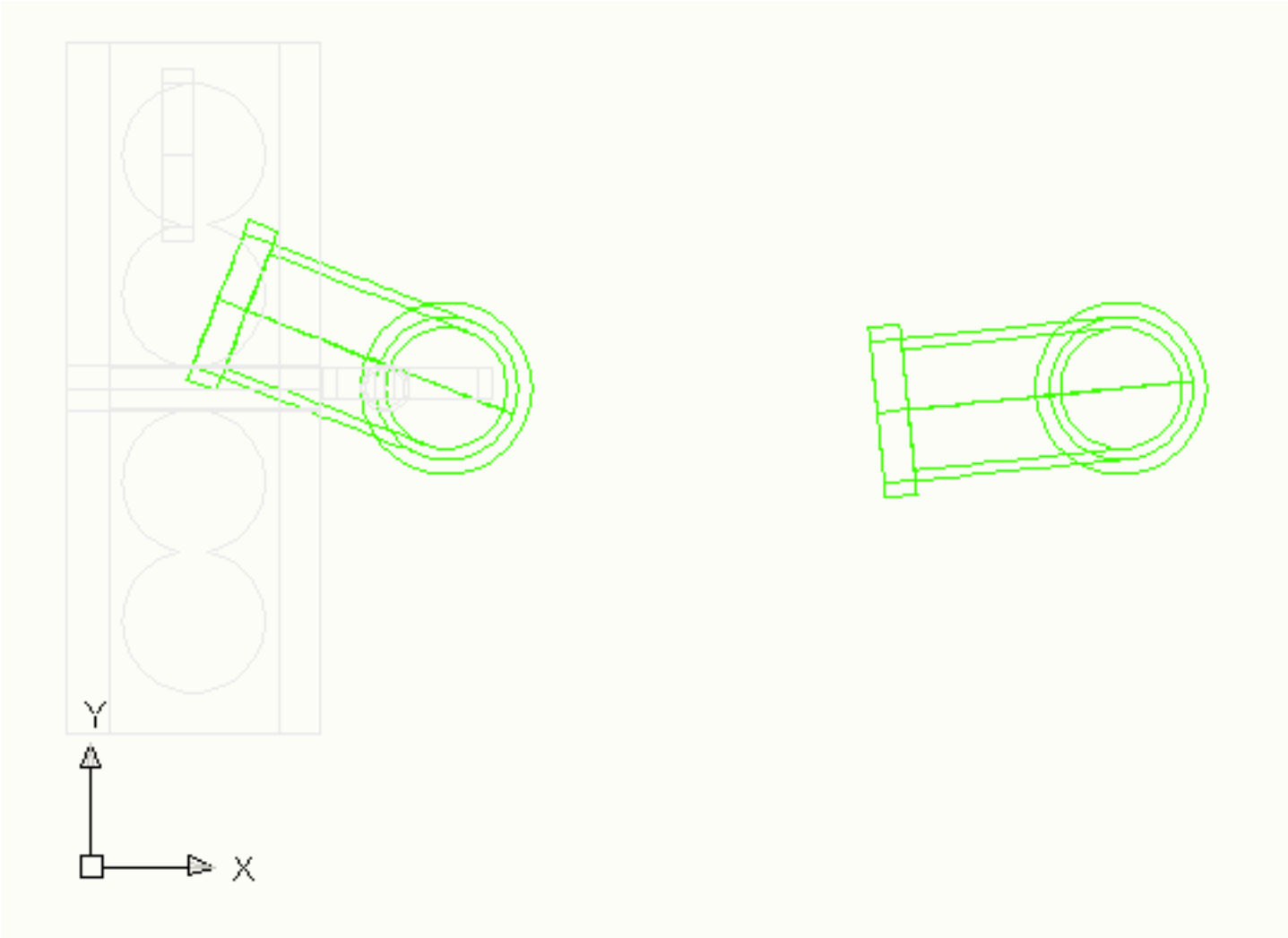
$sedtankelbow10_{origin}$ =

- x: $tank_{origin0}$ - L_{Sed} + $W_{Channel}$ + T{~}ChannelWall
- y: $tank_{origin1}$ + $W_{Sed}/2$ - ElbowRadius($ND_{SedLaunder}$)

- z: $tank_{origin2}$ + H_{Sed} - $H_{Channel}$ - T_{Mp}

"z" - specifies axis that object will be rotated about.

$sedtankpipe7_{angle}$ = -(rad/deg) * atan(((3*L_{Sed}/8 - $W_{Channel}$/2)/ ((W_{Sed}/2 - outerradius($ND_{SedLaunder}$))/4) + outerradius($ND_{SedLaunder}$))$^{-1}$)



Top View

rotate17 - [Rotate$_{3d}$] turns the object based on a given axis and degree angle.

rotate17 <- rotate$_{3d}$(p1,sedtankpipe11$_{origin}$,"z".sedtankpipe5$_{angle}$)

Note: p1 is a dummy variable used only in the program help section to designate the matrix below.

p1 =

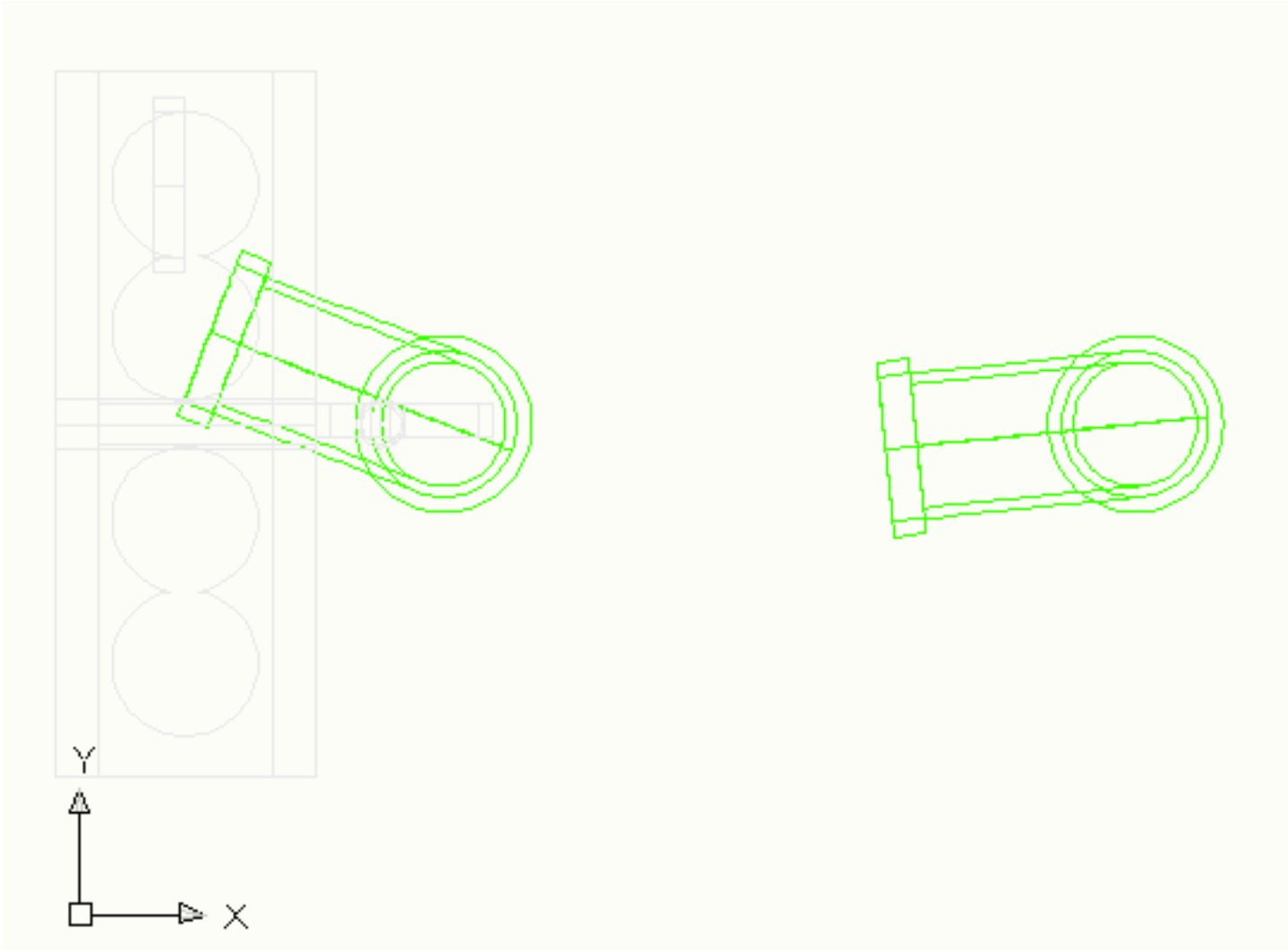
- x: sedtankpipe10$_{origin1}$ - outerradius($ND_{SedPipeEntrance}$)
- y: sedtankpipe11$_{origin2}$
- z: sedtankpipe12$_{origin0}$

sedtankpipe11$_{origin}$ =

- x: $tank_{origin0}$ - (3*L_{Sed})/8
- y: $tank_{origin1}$ + W_{Sed}/2
- z: $tank_{origin2}$ + $S_{SedInlet}$

"z" - specifies axis that object will be rotated about.

sedtankpipe5{~}angle - specifies rotation angle.



Top View

rotate18 - Rotate$_{3d}$ turns the object based on a given axis and degree angle.

rotate18 <- rotate$_{3d}$(p1,sedtankpipe12$_{origin}$,"z".sedtankpipe8$_{angle}$)

Note: p1 is a dummy variable used only in the program help section to designate the matrix below.

p1 =

- x: sedtankpipe12$_{origin0}$
- y: sedtankpipe12$_{origin1}$ - outerradius(ND$_{SedPipeEntrance}$)
- z: sedtankpipe12$_{origin2}$

sedtankpipe12$_{origin}$ =

- x: tank$_{origin0}$ - (L_{Sed})/8
- y: tank$_{origin1}$ + W_{Sed}/2
- z: tank$_{origin2}$ + $S_{SedInlet}$

"z" - specifies axis that object will be rotated about.

sedtankpipe8{~}angle - specifies rotation angle.
layerset - Layer$_{set}$ selects the layer "0".

layerset <- $layer_{set}$("0")

layerfreeze5 - $Layer_{freeze}$ locks the layer "elbowa" so that it cannot be edited.

layerfreeze5 <- $layer_{freeze}$("elbowa")

layer6 - $Layer_{freeze}$ locks the layer "elbowb" so that it cannot be edited.

layer6 <- $layer_{new}$("elbowb",green)

elbow9 - Calls the Elbow Program to create an elbow.

elbow9 <- elbow($sedtankelbow9_{origin}$,$ND_{SedLaunder}$,$EN_{PipeSpec}$)

$sedtankelbow9_{origin}$ =

- x: $tank_{origin0}$ - L_{Sed} + $W_{Channel}$ + $T_{ChannelWall}$ + ElbowRadius($ND_{SedLaunder}$~)
- y: $tank_{origin1}$ + W_{Sed}/2 - ElbowRadius($ND_{SedLaunder}$)
- z: $tank_{origin2}$ + HW_{Sed} - $H_{SedAbove}$/2 - ElbowRadius($ND_{SedLaunder}$)

$ND_{SedPipeEntrance}$ = The nominal diameter of the pipe. This value along with the pipe schedule is used to determine other actual dimensions of the elbow

$EN_{PipeSpec}$ = The enumerated pipe schedule type. Each schedule of pipe is represented by a specific number within our code.

rotate19 - $Rotate_{3d}$ turns the object based on a given axis and degree angle.

rotate19 <- $rotate_{3d}$(p1,p2,"z",90)

Note: p1 and p2 are dummy variables used only in the program help section to designate the matrix below.

p1 =

- x: $sedtankelbow9_{origin0}$
- y: $sedtankelbow9_{origin1}$ + ElbowRadius($ND_{SedLaunder}$)
- z: $sedtankelbow9_{origin2}$ + (innerD($ND_{SedLaunder}$,$EN_{PipeSpec}$)/2)

p2 =

- x: $sedtankelbow9_{origin0}$
- y: $sedtankelbow9_{origin1}$ + ElbowRadius($ND_{SedLaunder}$)
- z: $sedtankelbow9_{origin2}$

"z" - specifies axis that object will be rotated about.

90 - specifies rotation angle.

rotate20 - $Rotate_{3d}$ turns the object based on a given axis and degree angle.

rotate20 <- $rotate_{3d}$(p1,p2,"x",90)

Note: p1 and p2 are dummy variables used only in the program help section to designate the matrix below.

p1 =

- x: $sedtankelbow9_{origin0}$
- y: $sedtankelbow9_{origin1}$ + ElbowRadius($ND_{SedLaunder}$)
- z: $sedtankelbow9_{origin2}$ + (innerD($ND_{SedLaunder}$,$EN_{PipeSpec}$)/2)

p2 =

- x: $sedtankelbow9_{origin0}$

- y: sedtankelbow9$_{origin1}$ + ElbowRadius(ND$_{SedLaunder}$)
- z: sedtankelbow9$_{origin2}$

"x" - specifies axis that object will be rotated about.

90 - specifies rotation angle.

elbow10 - Calls the Elbow Program to create an elbow.

elbow10 <- elbow(sedtankelbow10$_{origin}$,ND$_{SedLaunder}$,EN$_{PipeSpec}$)

sedtankelbow10$_{origin}$ =

- x: tank$_{origin0}$ - L$_{Sed}$ + W$_{Channel}$ + T{~}ChannelWall
- y: tank$_{origin1}$ + W$_{Sed}$/2 - ElbowRadius(ND$_{SedLaunder}$)
- z: tank$_{origin2}$ + H$_{Sed}$ - H$_{Channel}$ - T$_{Mp}$

ND$_{SedPipeEntrance}$ = The nominal diameter of the pipe. This value along with the pipe schedule is used to determine other actual dimensions of the elbow

EN$_{PipeSpec}$ = The enumerated pipe schedule type. Each schedule of pipe is represented by a specific number within our code.

rotate21 - Rotate$_{3d}$ turns the object based on a given axis and degree angle.

rotate21 <- rotate$_{3d}$(p1,p2,"x",-90)

Note: p1 and p2 are dummy variables used only in the program help section to designate the matrix below.

p1 =

- x: sedtankelbow10$_{origin0}$
- y: sedtankelbow10$_{origin1}$ + ElbowRadius(ND$_{SedLaunder}$)
- z: sedtankelbow10$_{origin2}$ + (innerD(ND$_{SedLaunder}$,EN$_{PipeSpec}$)/2)

p2 =

- x: sedtankelbow10$_{origin0}$
- y: sedtankelbow10$_{origin1}$ + ElbowRadius(ND$_{SedLaunder}$)
- z: sedtankelbow10$_{origin2}$

"x" - specifies axis that object will be rotated about.

-90 - specifies rotation angle.

layerset - Layer$_{set}$ selects the layer "0".

layerset <- layer$_{set}$("0")

layerfreeze6 - Layer$_{freeze}$ locks the layer "elbowb" so that it cannot be edited.

layerfreeze6 <- layer$_{freeze}$("elbowb")

This page last changed on Nov 06, 2008 by ar329.

Sedimentation Tank Arraying Script

layerthaw1 - $Layer_{thaw}$ unlocks the layer "slopes" so that it can be edited.

layerthaw1 <- $layer_{thaw}$("slopes")

layerthaw2 - $Layer_{thaw}$ unlocks the layer "channel" so that it can be edited.

layerthaw2 <- $layer_{thaw}$("channel")

layerthaw2e - $Layer_{thaw}$ unlocks the layer "echannel" so that it can be edited.

layerthaw2e <- $layer_{thaw}$("echannel")

layerthaw3 - $Layer_{thaw}$ unlocks the layer "pipe" so that it can be edited.

layerthaw3 <- $layer_{thaw}$("pipe")

layerthaw4 - $Layer_{thaw}$ unlocks the layer "elbow" so that it can be edited.

layerthaw4 <- $layer_{thaw}$("elbow")

layerthaw4a - $Layer_{thaw}$ unlocks the layer "elbowa" so that it can be edited.

layerthaw4a <- $layer_{thaw}$("elbowa")

layerthaw4b - $Layer_{thaw}$ unlocks the layer "elbowb" so that it can be edited.

layerthaw4b <- $layer_{thaw}$("elbowb")

layerthaw5 - $Layer_{thaw}$ unlocks the layer "manifold" so that it can be edited.

layerthaw5 <- $layer_{thaw}$("manifold")

layerthaw6 - $Layer_{thaw}$ unlocks the layer "sludgepipe" so that it can be edited.

layerthaw6 <- $layer_{thaw}$("sludgepipe")

layerthaw7 - $Layer_{thaw}$ unlocks the layer "lamina" so that it can be edited.

layerthaw7 <- $layer_{thaw}$("lamina")

layerthaw8 - $Layer_{thaw}$ unlocks the layer "inletpipes" so that it can be edited.

layerthaw8 <- $layer_{thaw}$("inletpipes")

layerthaw8is - $Layer_{thaw}$ unlocks the layer "inletslopes" so that it can be edited.

layerthaw8is <- $layer_{thaw}$("inletslopes")

viewtop - rotates the workspace so that the object is viewed from the top.

viewtop <- viewtop1

layerfreeze7 - $Layer_{freeze}$ locks the layer "pipe" so that it cannot be edited.

layerfreeze7 <- $layer_{freeze1}$("pipe")

layerfreeze8 - $Layer_{freeze}$ locks the layer "elbow" so that it cannot be edited.

layerfreeze8 <- layer$_{freeze}$("elbow")

layerfreeze8a - Layer$_{freeze}$ locks the layer "elbowa" so that it cannot be edited.

layerfreeze8a <- layer$_{freeze}$("elbowa")

layerfreeze8b - Layer$_{freeze}$ locks the layer "elbowb" so that it cannot be edited.

layerfreeze8b <- layer$_{freeze}$("elbowb")

layerfreeze9 - Layer$_{freeze}$ locks the layer "sludgepipe" so that it cannot be edited.

layerfreeze9 <- layer$_{freeze}$("sludgepipe")

layerfreeze10 - Layer$_{freeze}$ locks the layer "manifold" so that it cannot be edited.

layerfreeze10 <- layer$_{freeze}$("manifold")

layerfreeze12 - Layer$_{freeze}$ locks the layer "lamina" so that it cannot be edited.

layerfreeze12 <- layer$_{freeze}$("lamina")

layerfreeze13 - Layer$_{freeze}$ locks the layer "inletpipes" so that it cannot be edited.

layerfreeze13 <- layer$_{freeze}$("inletpipes")

layerfreeze14 - Layer$_{freeze}$ locks the layer "channel" so that it cannot be edited.

layerfreeze14 <- layer$_{freeze}$("channel")

layerfreeze15 - Layer$_{freeze}$ locks the layer "echannel" so that it cannot be edited.

layerfreeze15 <- layer$_{freeze}$("echannel")

layerfreeze15is - Layer$_{freeze}$ locks the layer "inletslopes" so that it cannot be edited.

layerfreeze15is <- layer$_{freeze}$("inletslopes")

bigunion - Union$_{allA}$ selects all the objects in the workspace and joins them into one single object.

bigunion <- union$_{allA}$

layerset - Layer$_{set}$ selects the layer "0".

layerset <- layer$_{set}$("0")

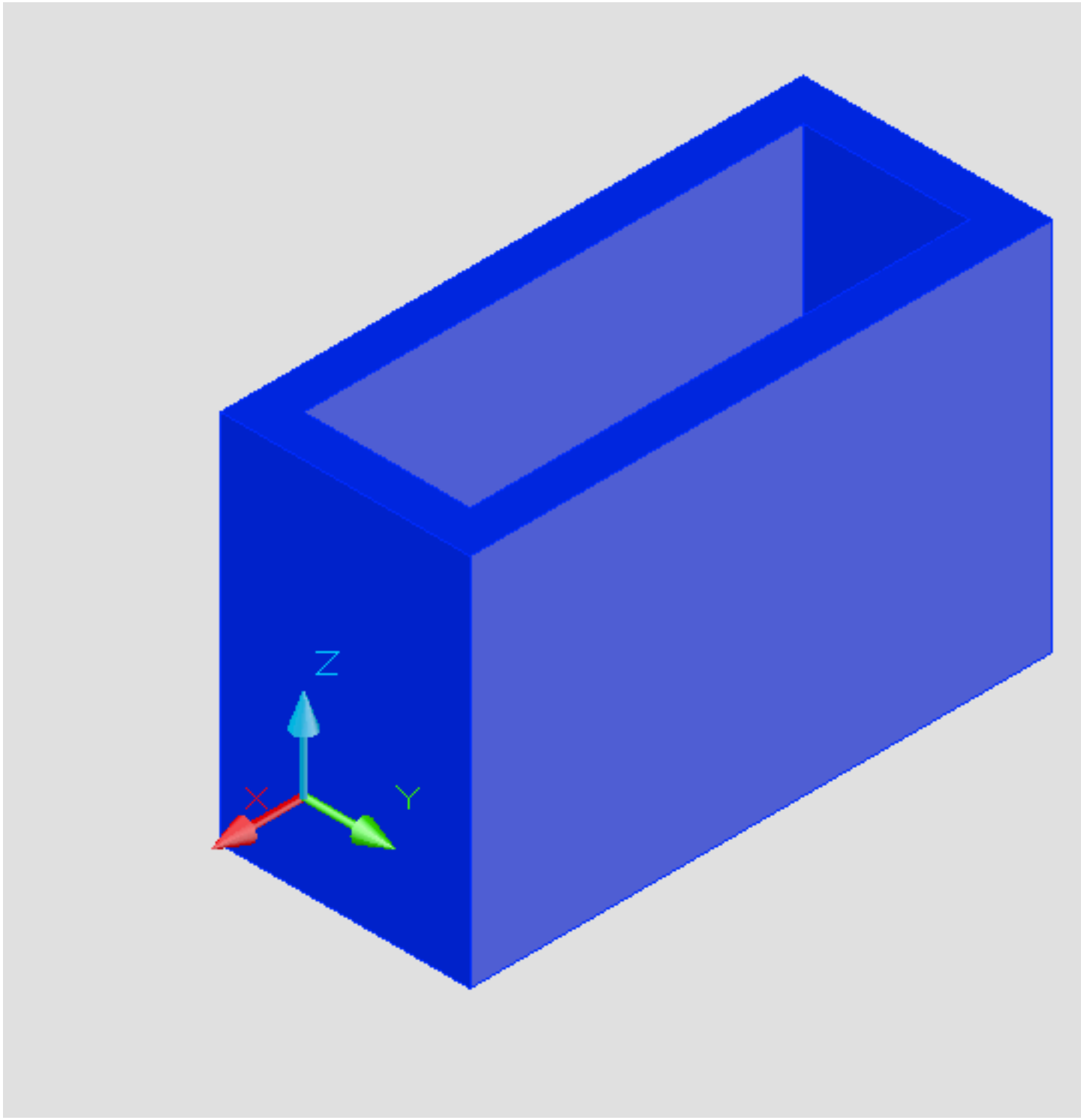
layerfreeze16 - Layer$_{freeze}$ locks the layer "slopes" so that it cannot be edited.

layerfreeze16 <- layer$_{freeze}$("slopes")

This page last changed on Dec 18, 2008 by ar329.

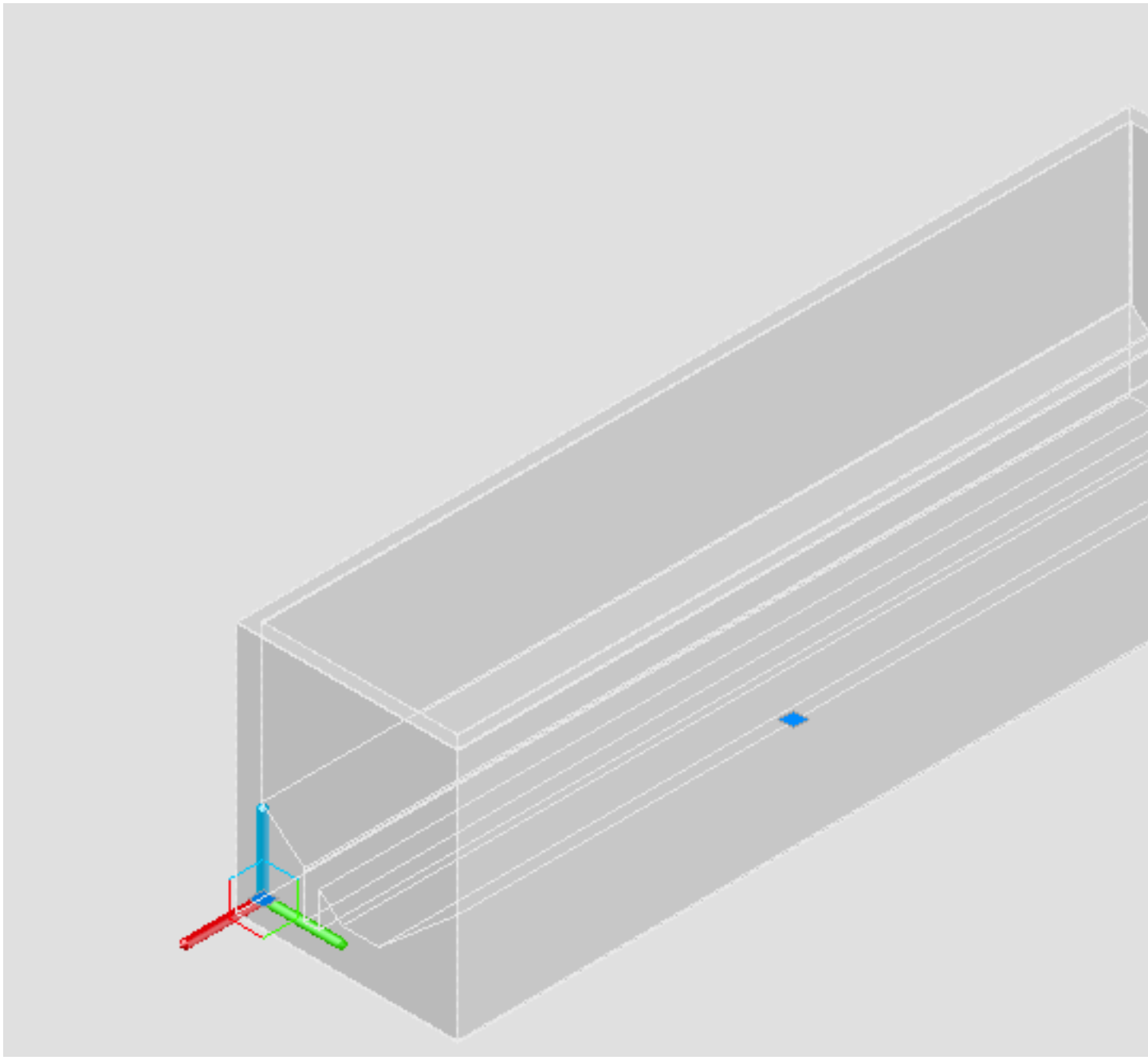
AutoCAD Sedimentation Tank Program

Sedimentation Tank Program Scripts



Northeast Isometric View

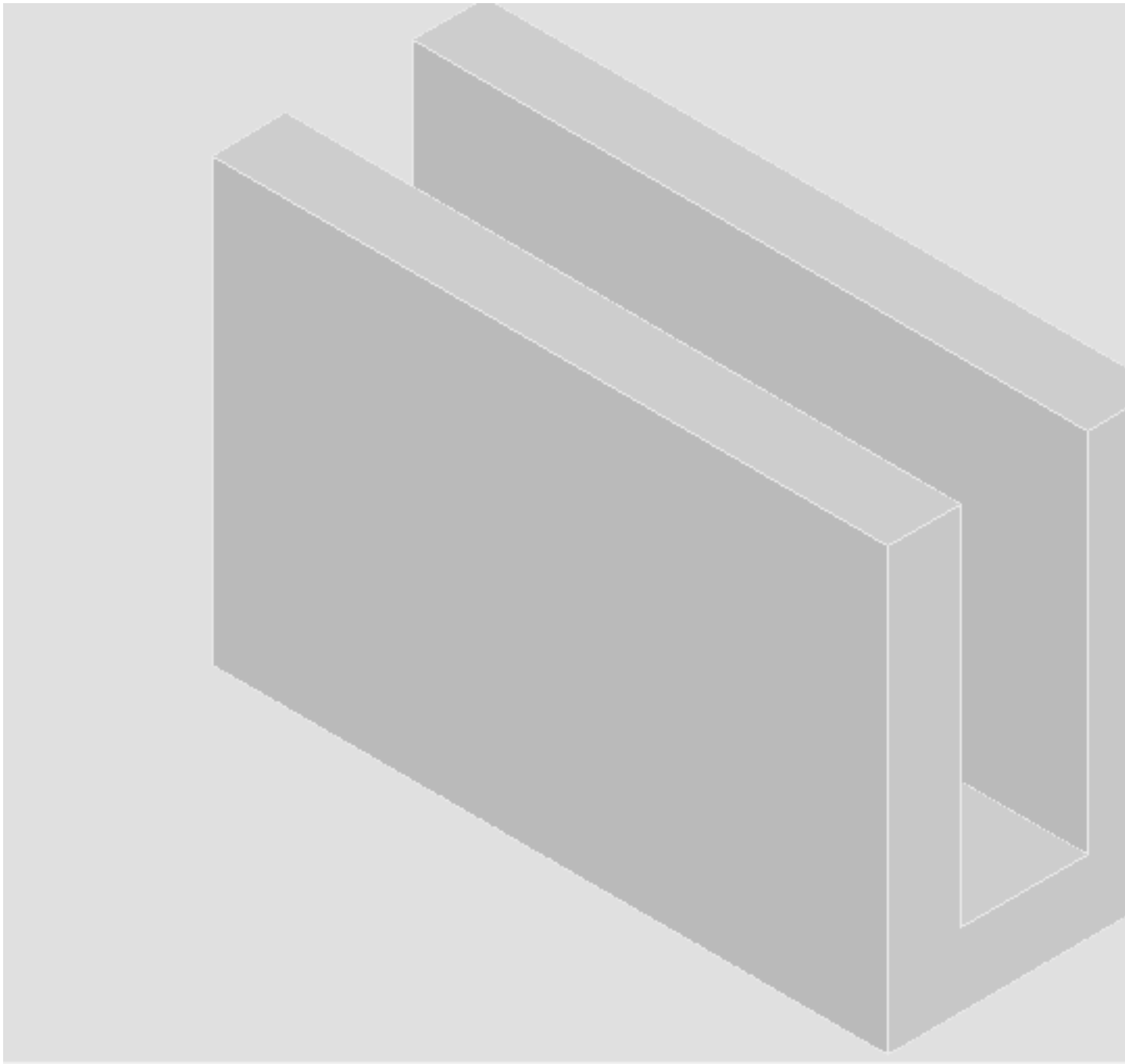
Sedtanktankscript (ready for initial review - Anastasia)



Northeast Isometric View

Sedtankslopescript (ready for initial review - Anastasia)

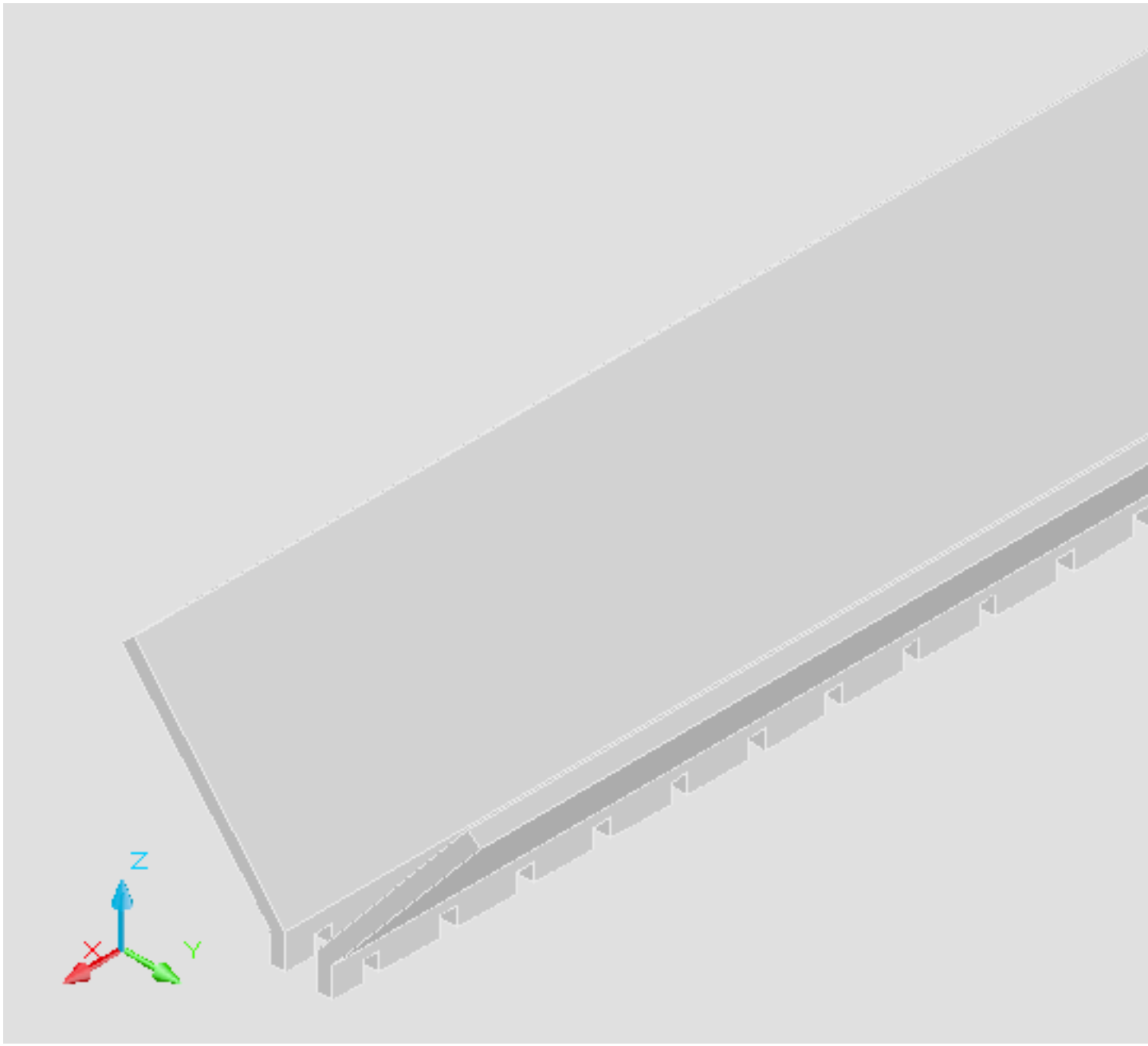
(add picture here of what this script draws)
Sedtankinletslopescript



Northeast Isometric View

Sedtankexitchannelscript- Ready for initial review (Anastasia)

(add picture here of what this script draws)
Sedtankinletchannelscript

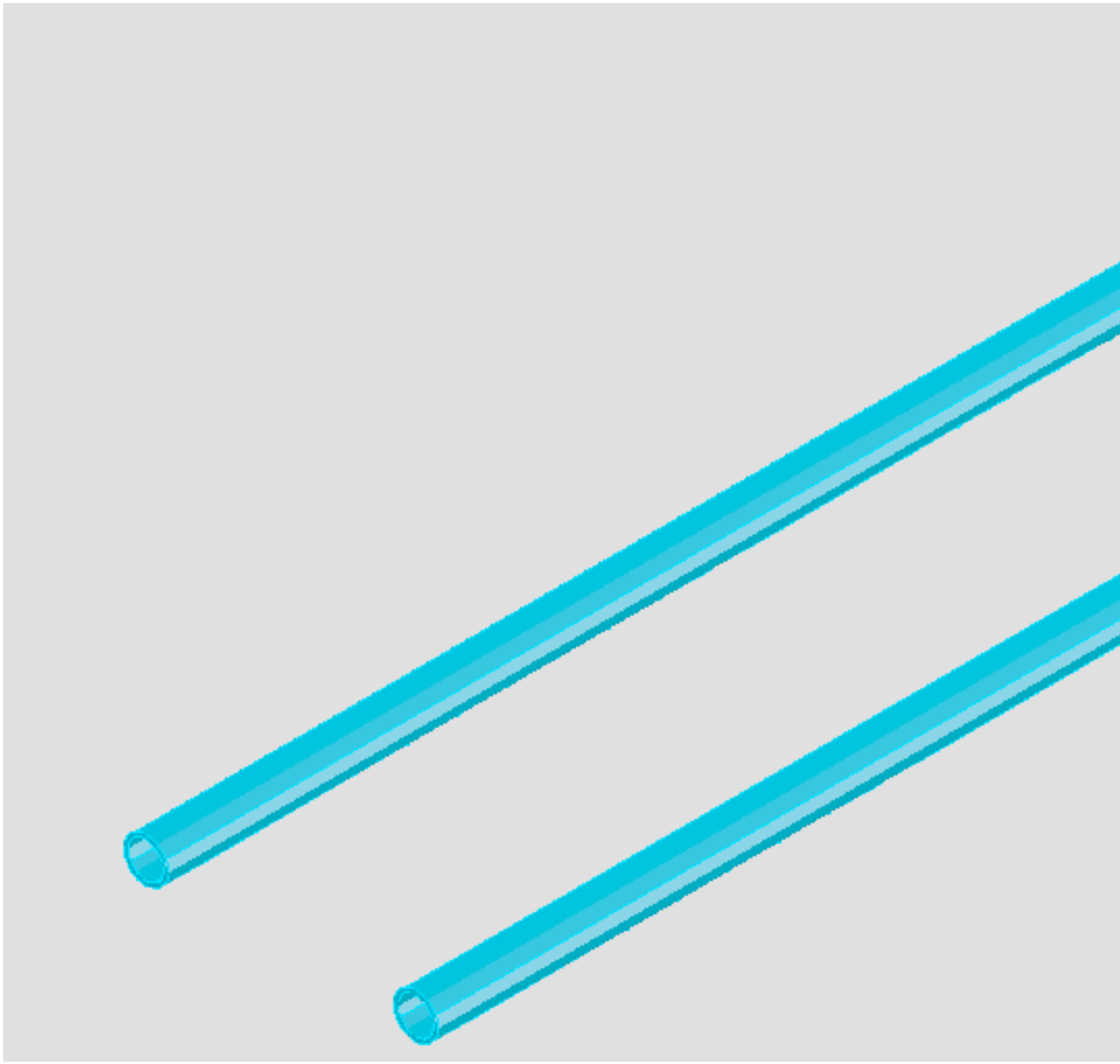


Northeast Isometric View

Sedtankinletpipescriptlayout1

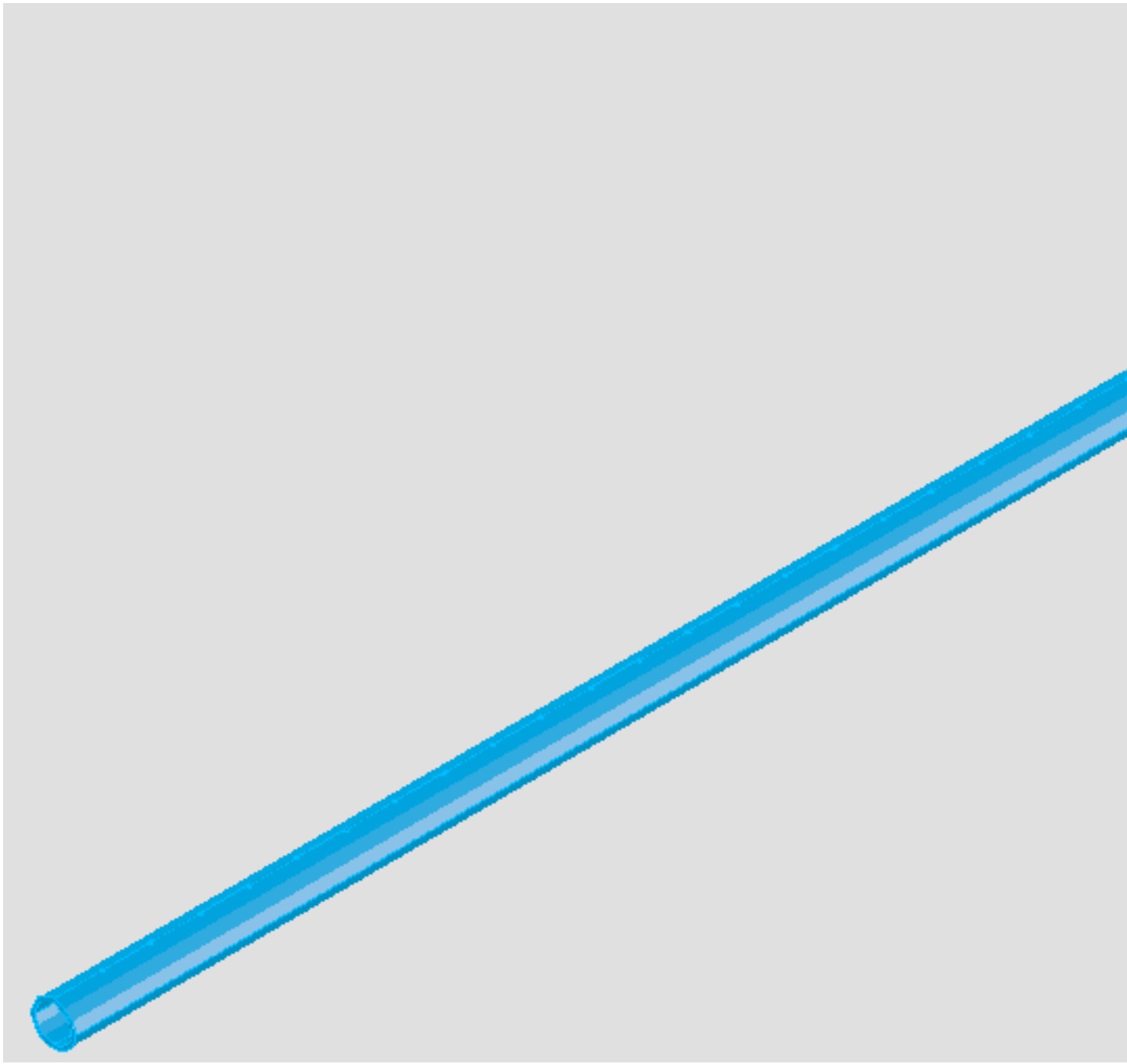
(add picture here of what this script draws)
Sedtankinletpipescript

(add picture here of what this script draws)
Sedtankelbowscript



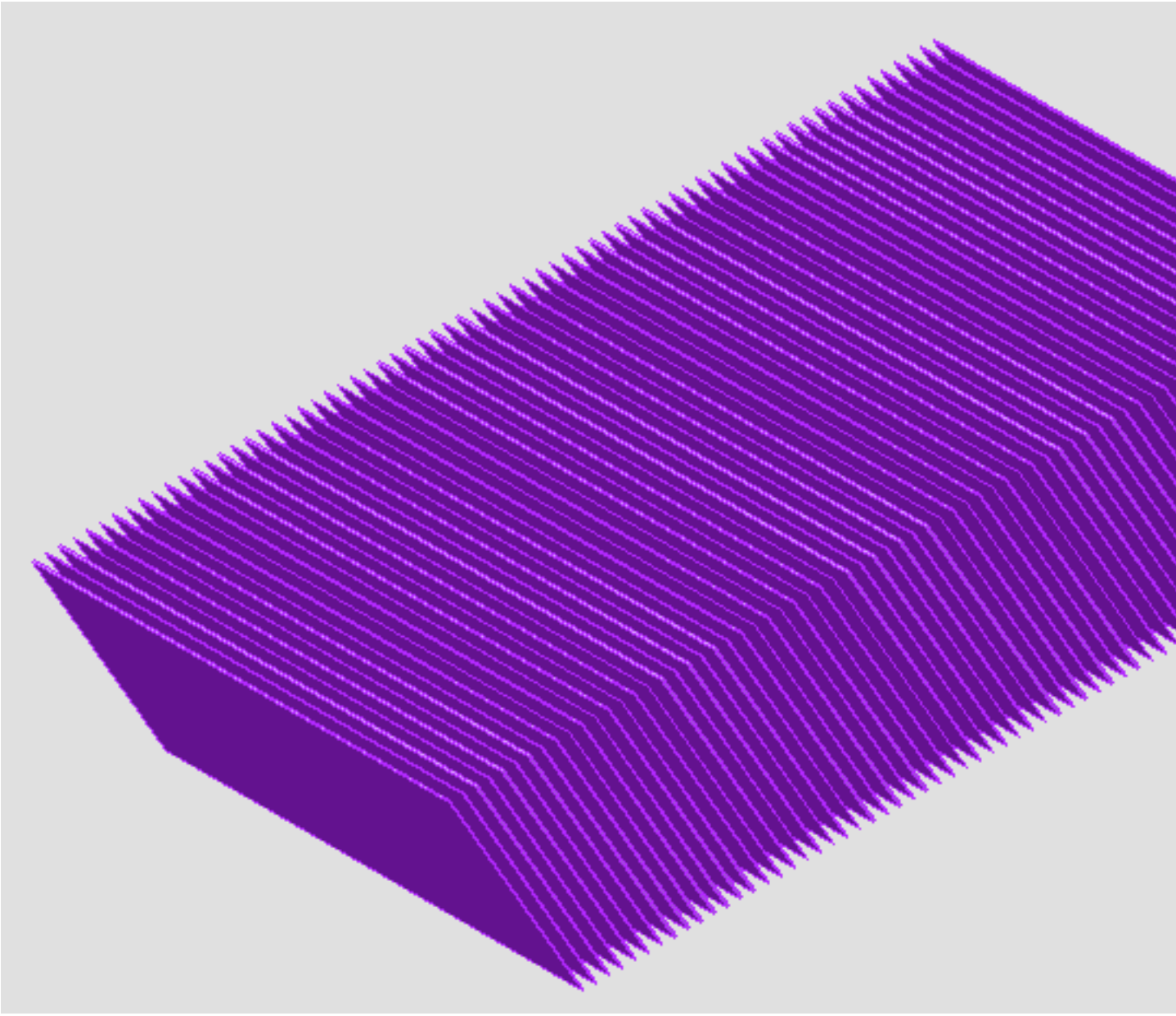
Northeast Isometric View

Sedtankmanifoldscript - ready for initial review (Anastasia)



Northeast Isometric View

(add picture here of what this script draws)
Sedtanksludgescript

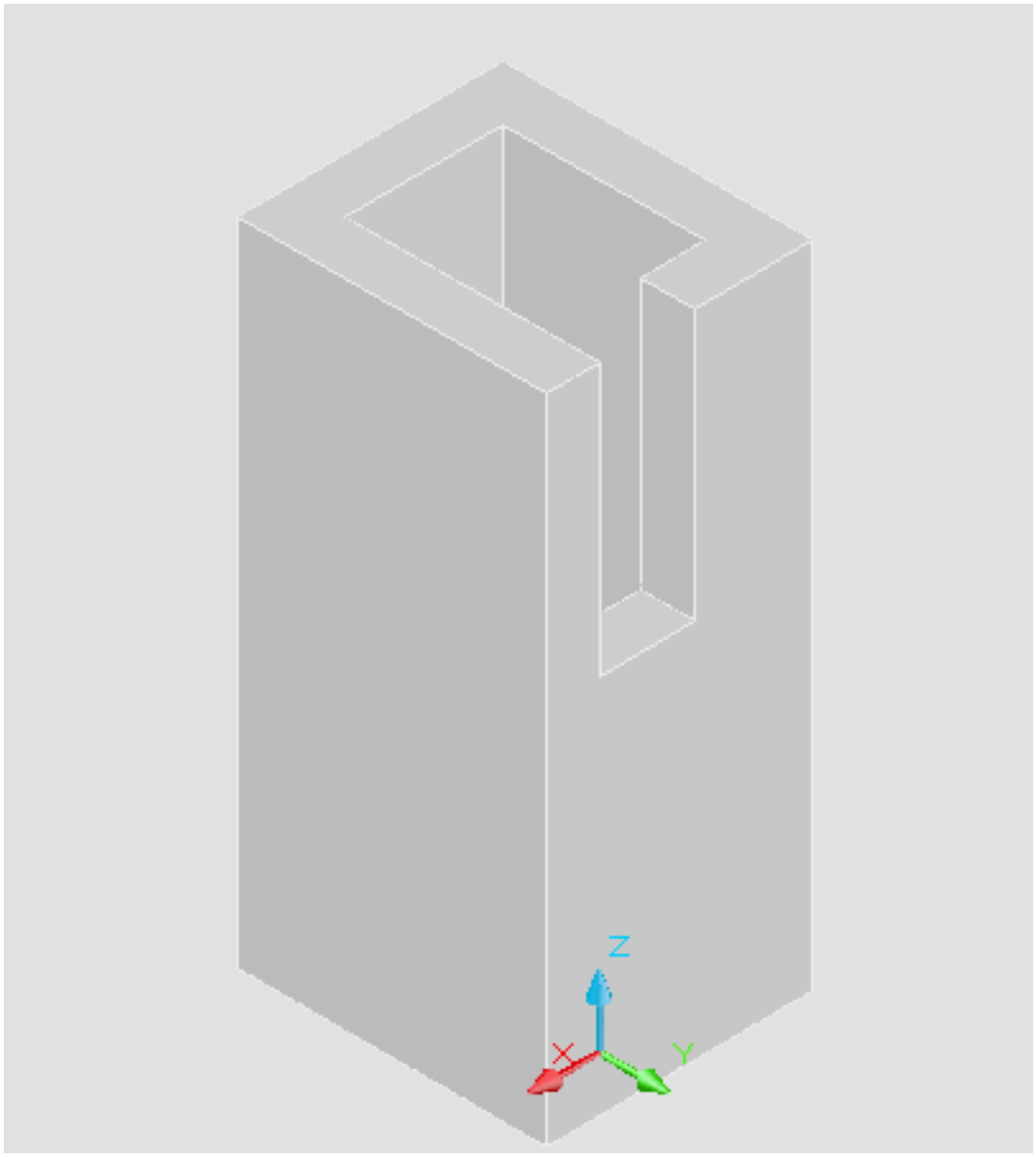


Northeast Isometric View

Sedtanklaminascript - ready for initial review (Anastasia)

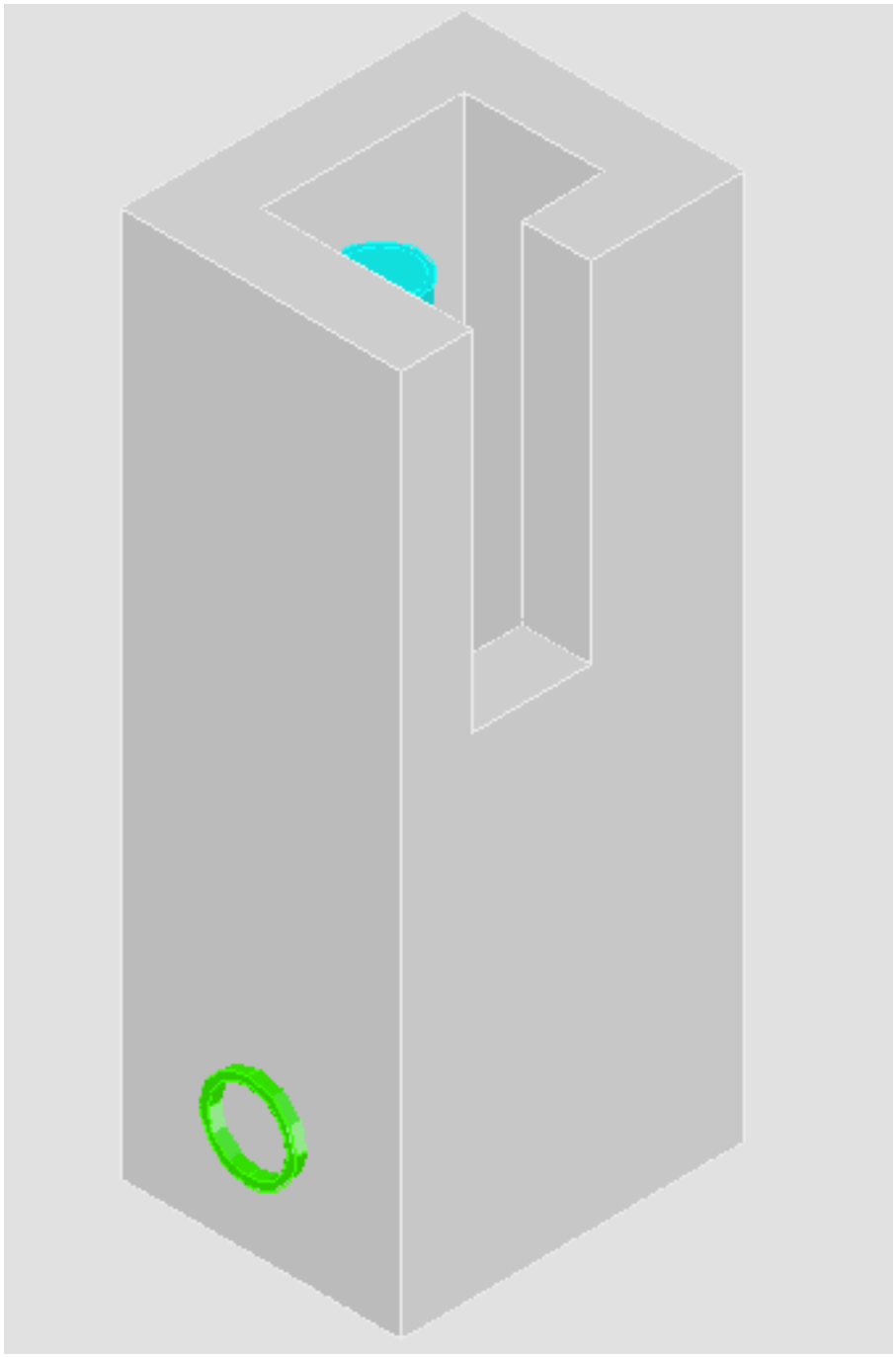
(add picture here of what this script draws)
Sedtankarrayscript (ready
for initial review - Anastasia)

(add picture here of what this script draws)
Sedtankopenchannelscript



Northeast Isometric View

Sedechanneltankscript - ready for initial review (Anastasia)

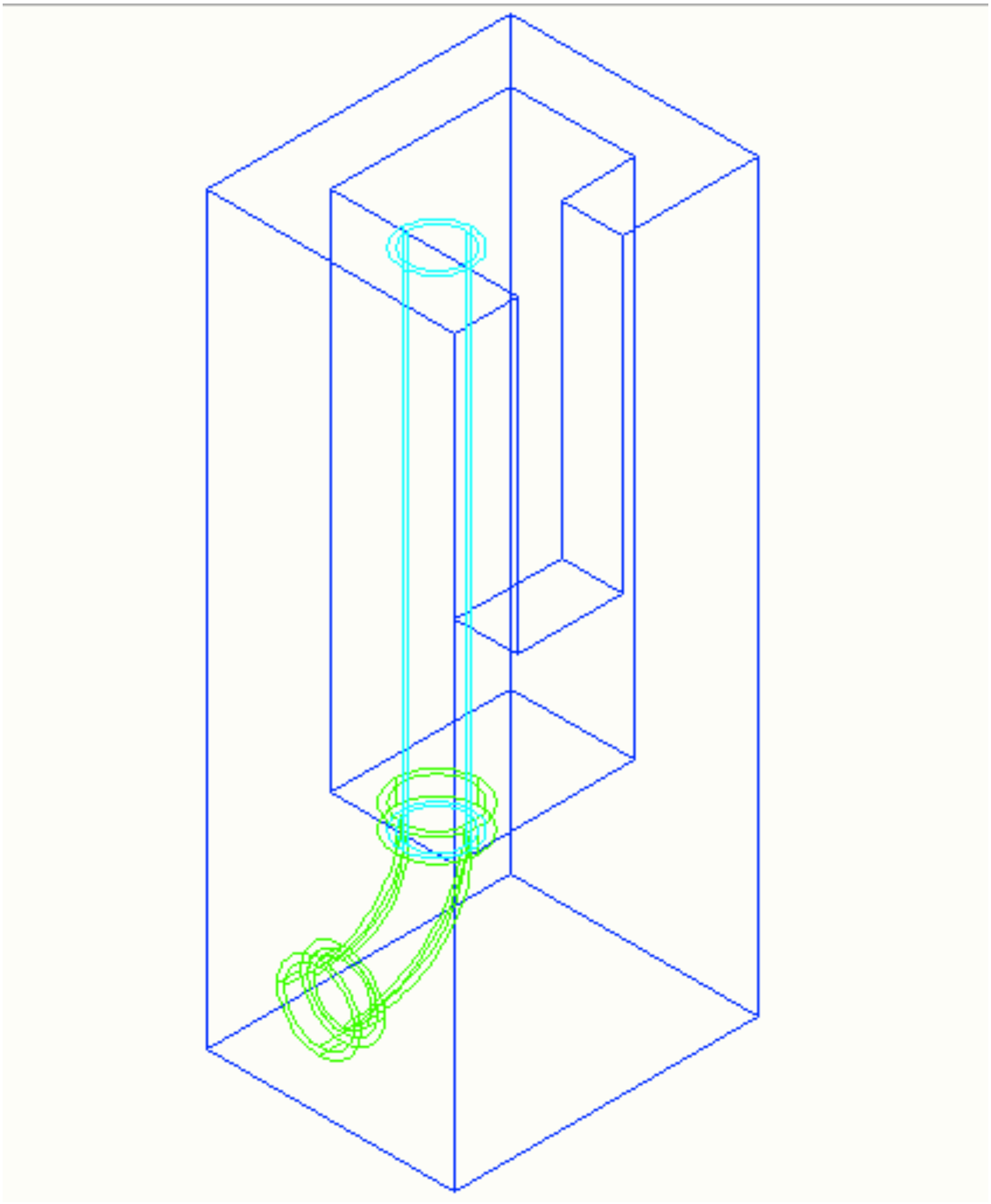


Northeast Isometric View

Sedichanneltankscript

This page last changed on Dec 18, 2008 by ar329.

Sedichanneltankscript



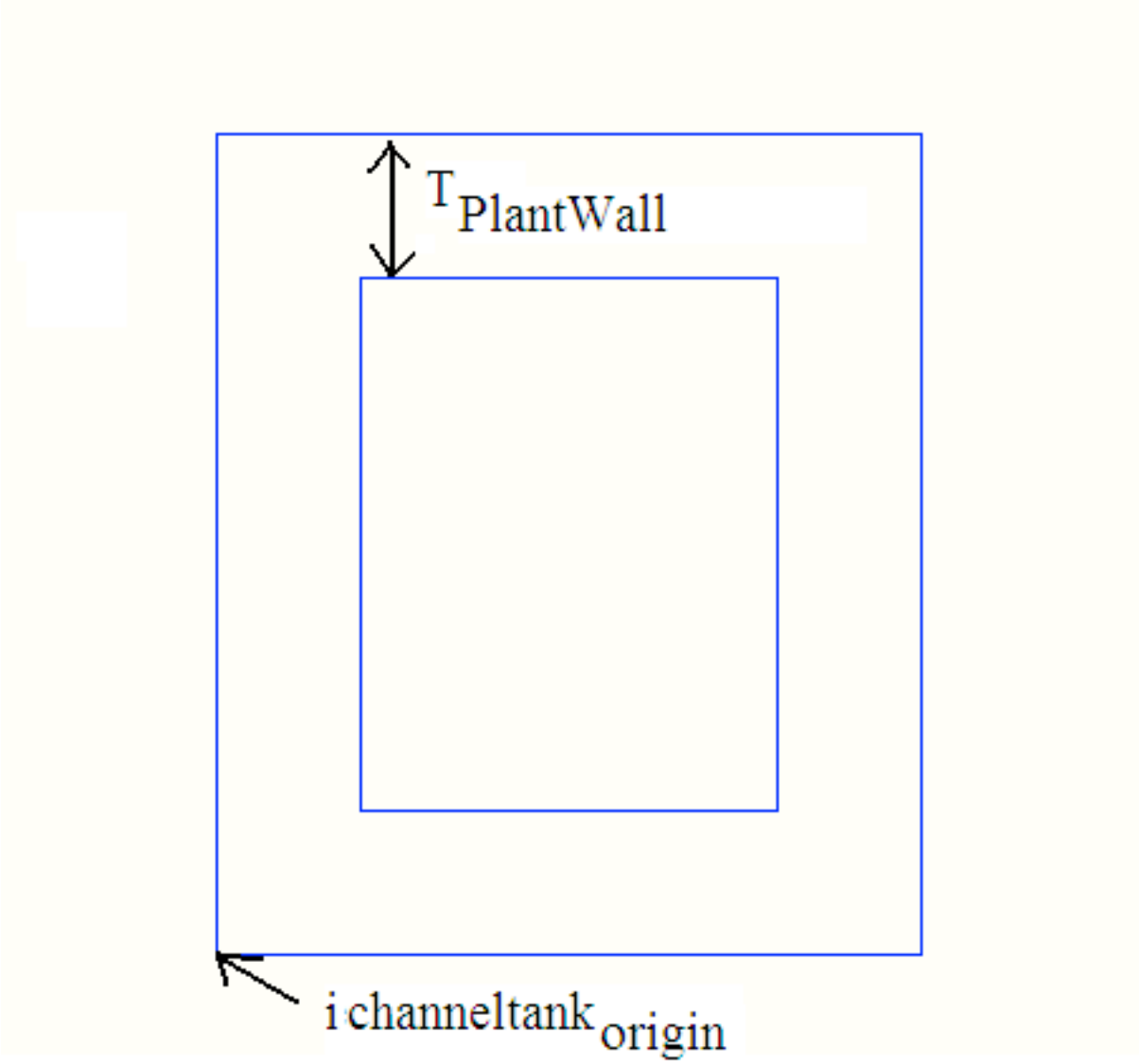
NorthEast Isometric View

viewtopi - Rotates workspace so that object is being viewed from the top.

viewtop <- viewtop1

layernewict - Layer$_{new}$ creates a new light grey layer, "ichanneltank."

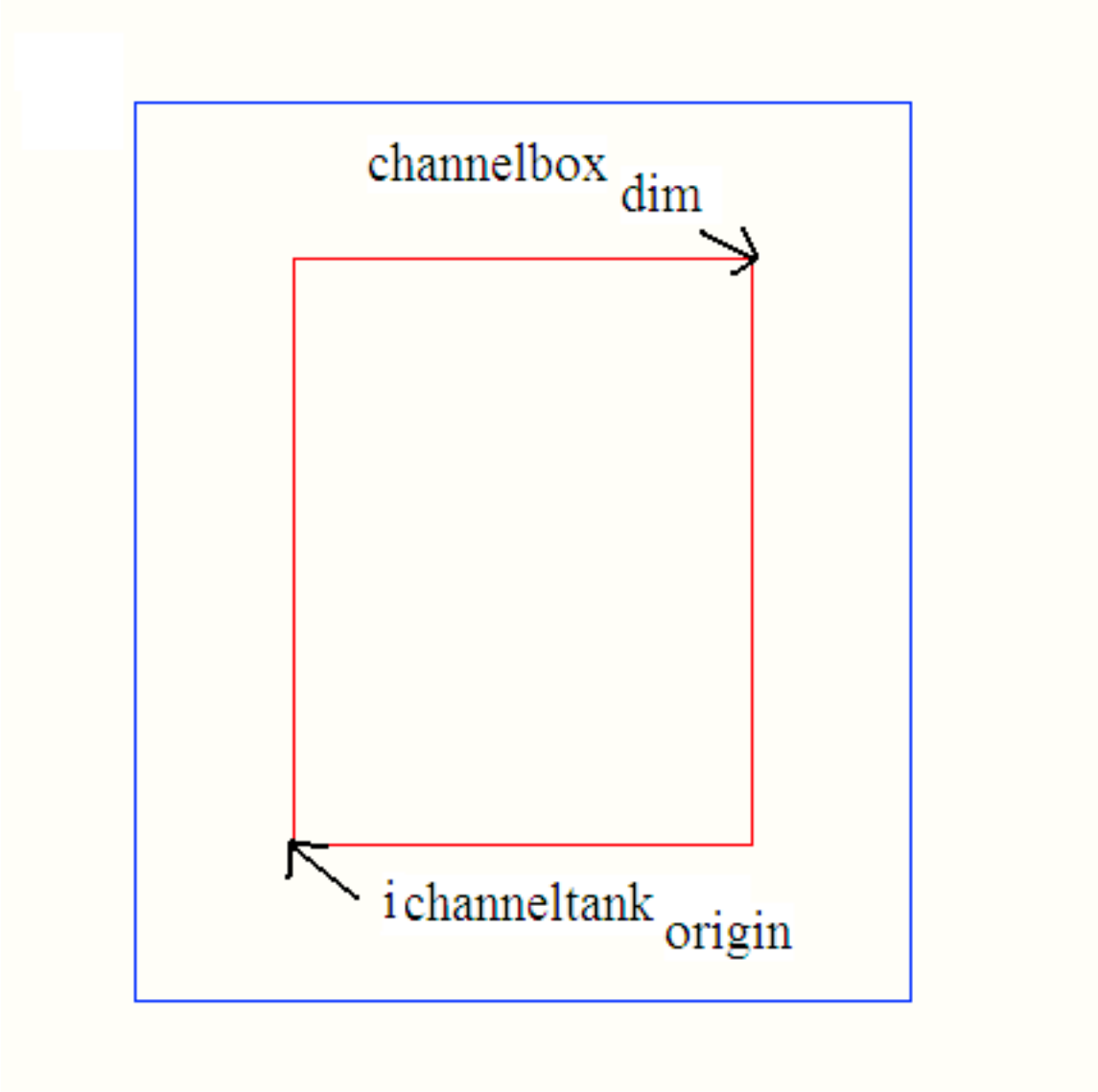
layernewict <- layer$_{new}$("ichanneltank",ltgrey)



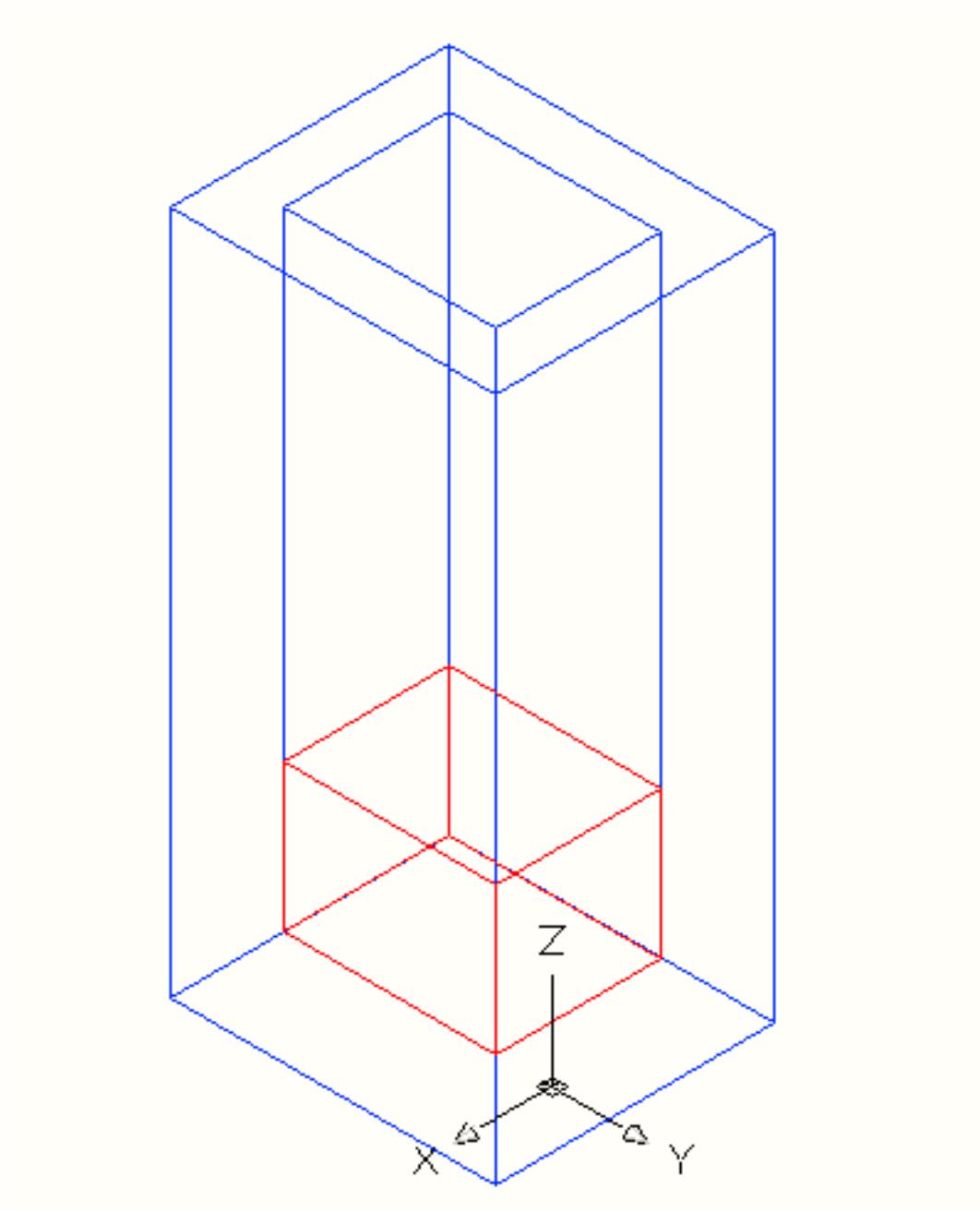
Top View

tankict - Calls the Tank Program to draw a tank.

tankict <- Tank($ichanneltank_{origin}$,$ichanneltank_{dim}$,$T_{PlantWall}$)



Top View



NorthEast Isometric View

boxictb - Creates a box based on two points.

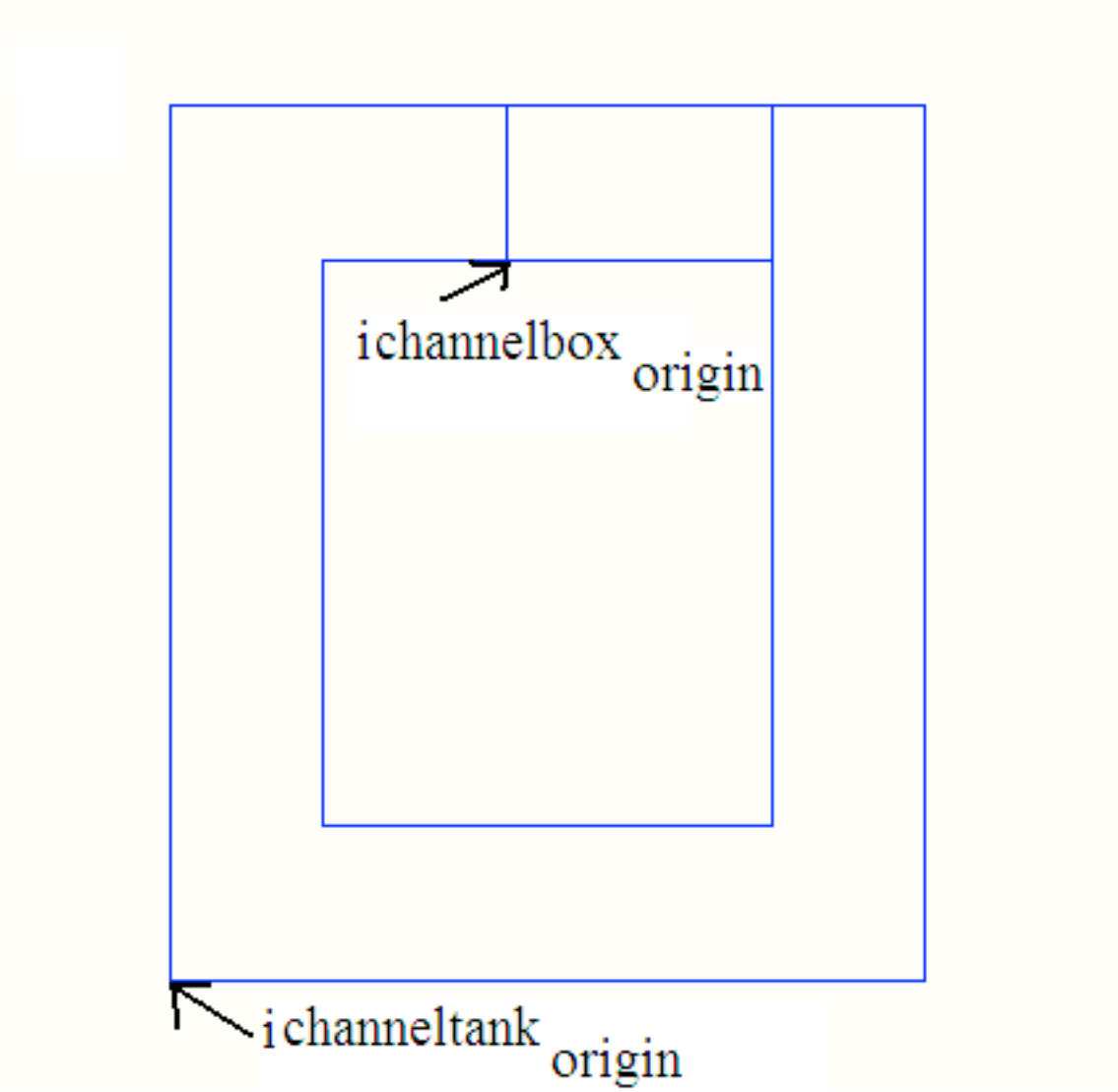
boxictb <- box($channelbox_{origin}$,$channelbox_{origin}$ + $channelbox_{dim}$)

$channelbox_{origin}$ =

- x: $tank_{origin0}$ - L_{Sed}
- y: $tank_{origin1}$
- z: $tank_{origin2}$ + H_{Sed} - $H_{Channel}$

$channelbox_{dim}$ =

- x: $W_{Channel}$
- y: $N_{SedTanks(W}$Sed~ + $T_{PlantWall}$)
- z: $H_{Channel}$



Top View

subtractict - SubtractD subtracts one object from the other based on two points.

subtractict <- subtractD($ichanneltank_{origin}$,$channelbox_{origin}$)

$channelbox_{origin}$ =

- x: $tank_{origin0}$ - L_{Sed}
- y: $tank_{origin1}$
- z: $tank_{origin2}$ + H_{Sed} - $H_{Channel}$

layerset - $Layer_{set}$ selects the layer "0".

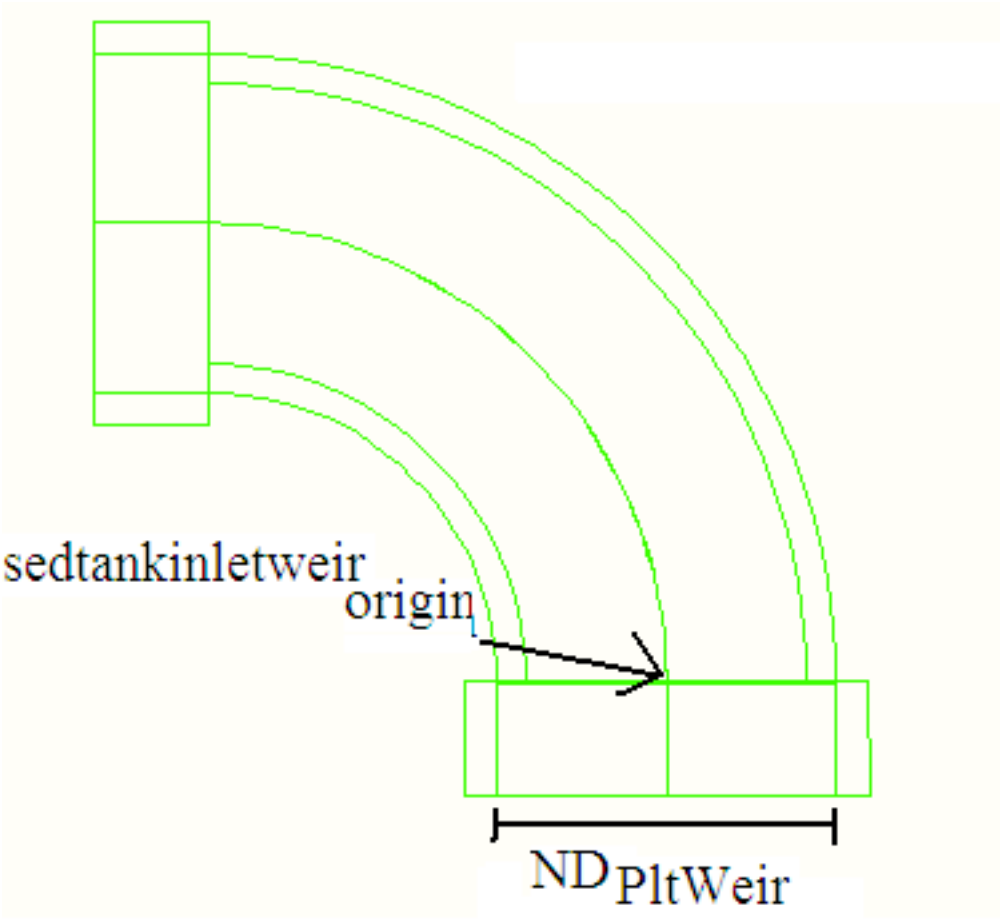
layerset <- $layer_{set}$("0")

layerfreezeict - $Layer_{freeze}$ locks the layer "ichanneltank" so that it cannot be edited."

layerfreezeict <- $layer_{freeze}$("ichanneltank")

layernewiwe - $Layer_{new}$ creates a new green layer, "inletweirelbow."

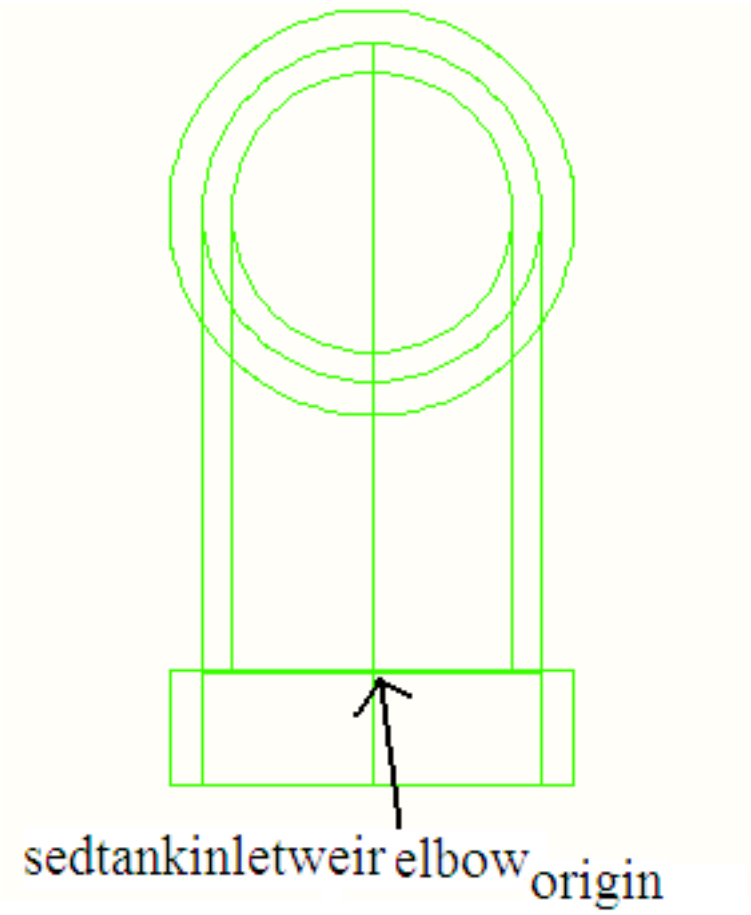
layernewiwe <- $layer_{new}$("inletweirelbow",green)



Top View

elbowiwe - Calls the Elbow Program to draw an elbow.

elbowiwe <- elbow(sedtankinletweirelbow$_{origin}$,ND$_{PltWeir}$,EN$_{Spec}$)



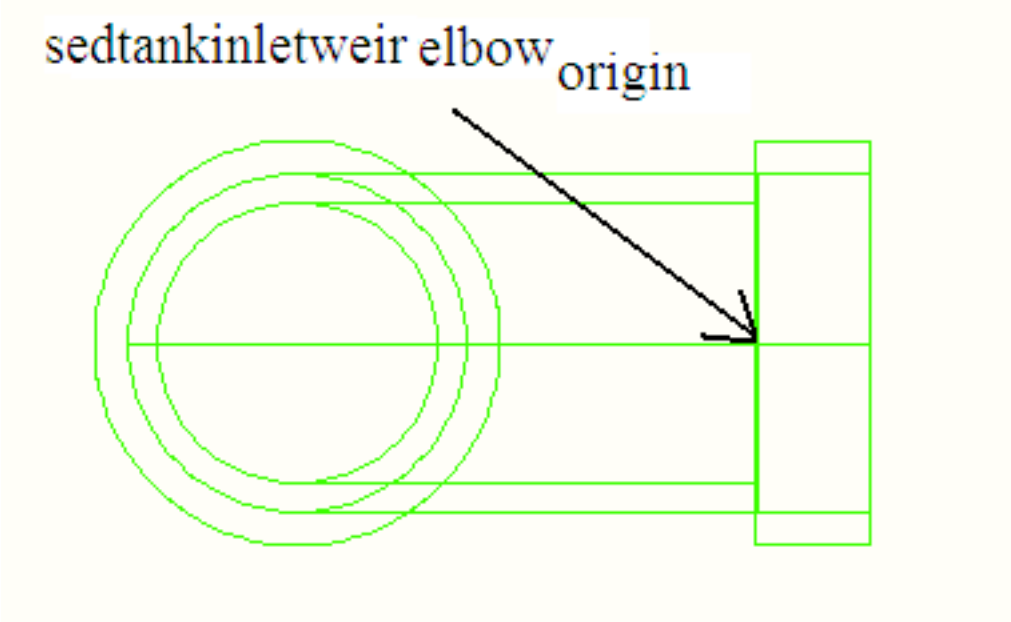
Top View

rotateiwe1 - Rotate$_{3d}$ rotates the selected object by the designated degrees.

rotateiwe1 <- rotate$_{3d}$(p1,sedtankinletweirelbow$_{origin}$,"y",90)

p1 =

- x: sedtankinletweirelbow$_{origin0}$ + ElbowRadius(ND$_{PltWeir}$)
- y: sedtankinletweirelbow$_{origin1}$
- z: sedtankinletweirelbow$_{origin2}$ - outerradius(ND$_{PltWeir}$)



Top View

rotateiwe2 - Rotate$_{3d}$ rotates the selected object by the designated degrees.

rotateiwe2 <- (p1,sedtankinletweirelbow$_{origin}$,"z",90)

p1 =

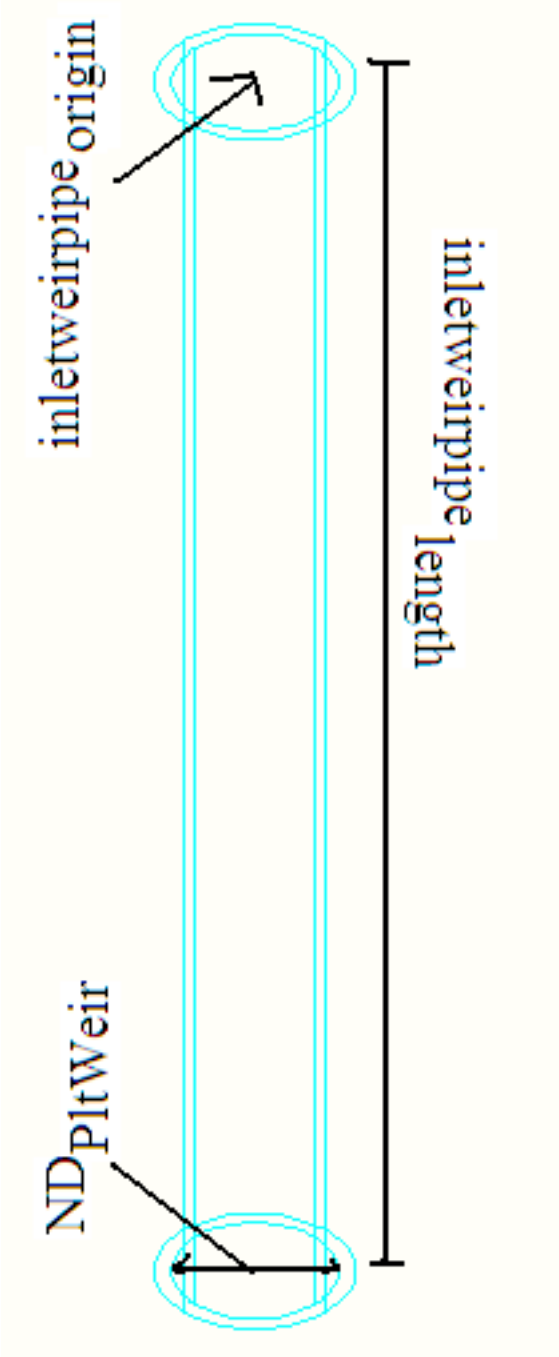
- x: sedtankinletweirelbow$_{origin0}$
- y: sedtankinletweirelbow$_{origin1}$ + ElbowRadius(NE$_{PltWeir}$) - outerradius(ND$_{PltWeir}$)
- z: sedtankinletweirelbow$_{origin2}$

layerfreezeiwe - Layer$_{freeze}$ locks the layer "inletweirelbow" so that it cannot be edited.

layerfreezeiwe <- layer$_{freeze}$("inletweirelbow")

layernewiwp - Layer$_{new}$ creates a new blue layer, "inletweir."

layernewiwp <- layer$_{new}$("inletweir",blue)



Top View

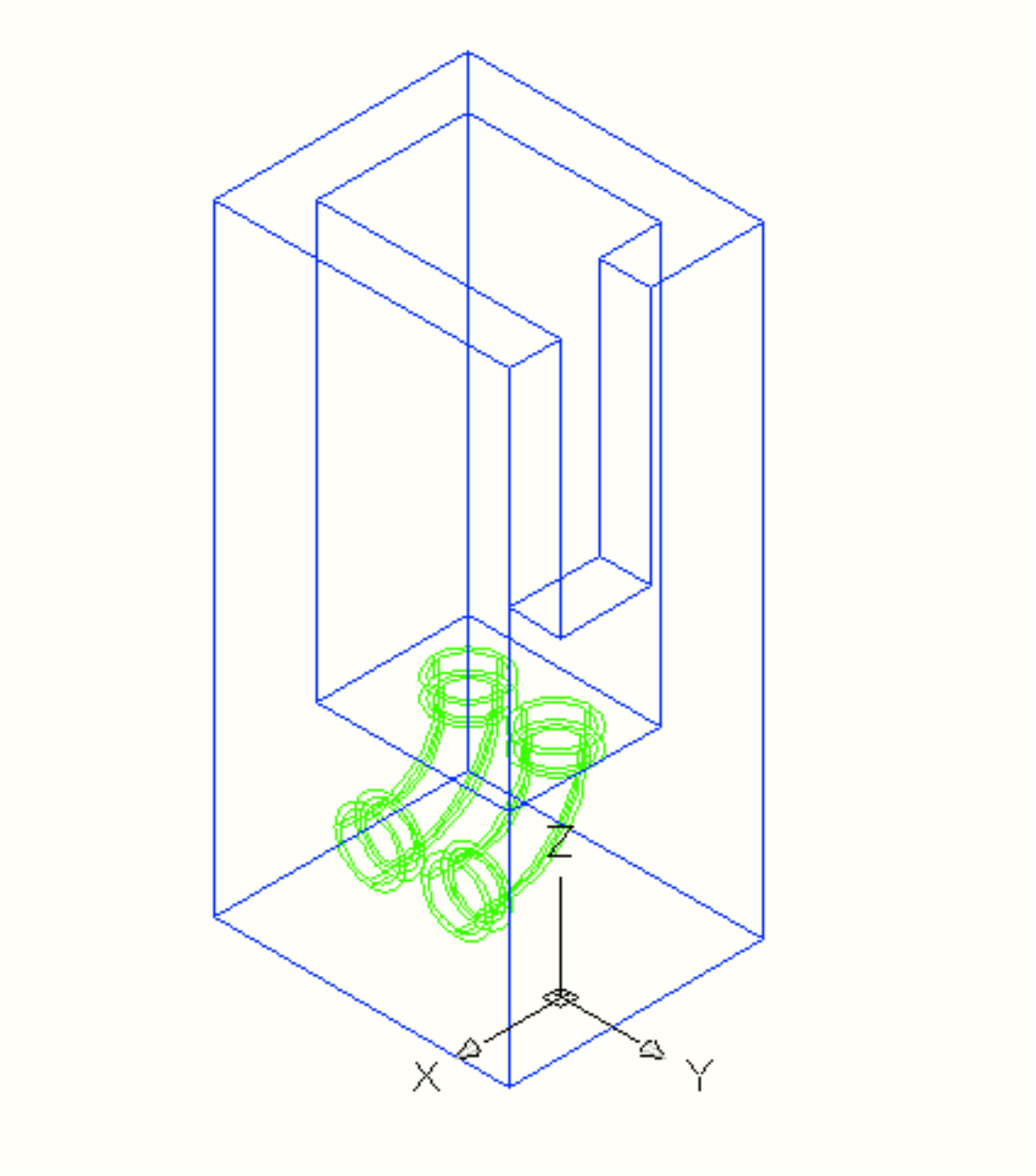
pipeiwp - Calls the Pipe Program to draw a pipe in the program.

pipeiwp <- Pipe($inletweirpipe_{origin}$,$ND_{PltWeir}$,$inletweirpipe_{length}$,$EN_{PipeSpec}$)
layerfreezeiwp - $Layer_{freeze}$ locks the layer "inletweir" so that it cannot be edited.

layerfreezeiwp <-$layer_{freeze}$("inletweir")

This page last changed on Dec 18, 2008 by ar329.

Sedechanneltankscript



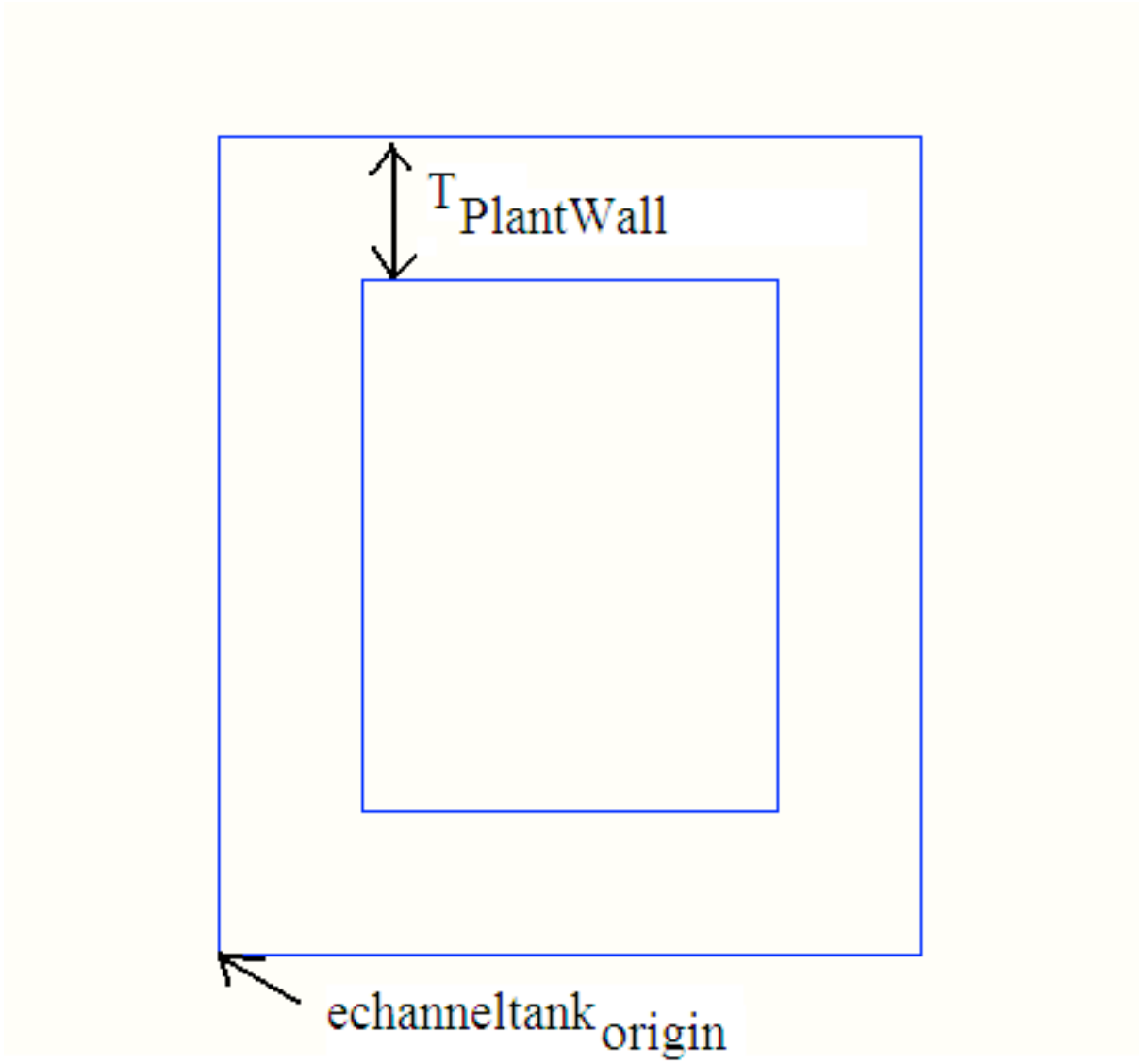
NorthEast Isometric View

viewtope - Rotates workspace so that object is being viewed from the top.

viewtope <- viewtop1

layernewect - Layer$_{new}$ creates a new light grey layer, "echanneltank."

layernewext <-layer$_{new}$("echanneltank",ltgrey)



Top View

tankect - Calls the Tank Program to draw a tank.

tankect <-Tank(echanneltank$_{origin}$,echanneltank$_{dim}$,$T_{PlantWall}$)

echannel$_{origin}$ =

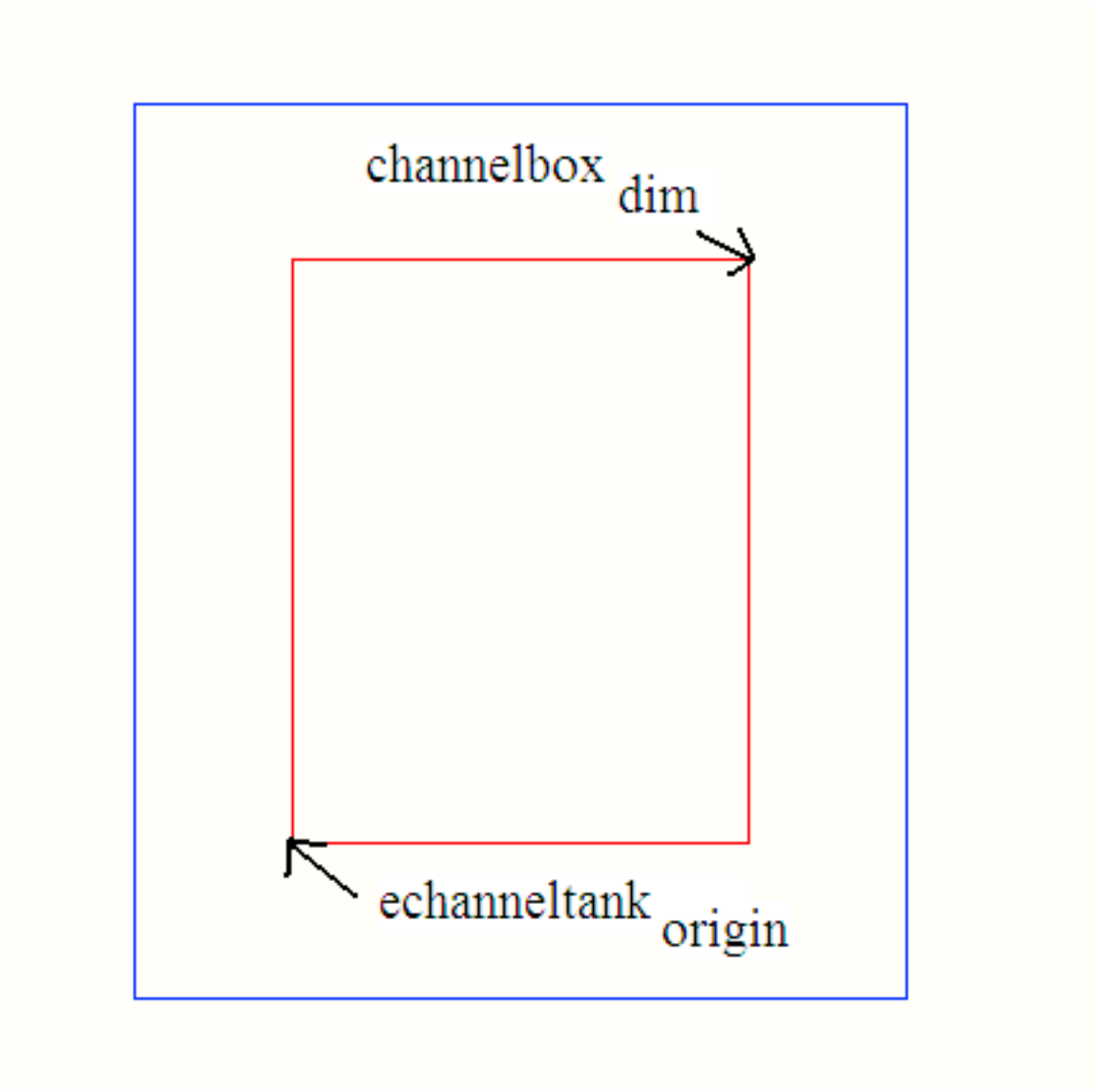
- if layout1:
  - x: 0
  - y: 0
  - z: 0
- if layout2:
  - x: tank$_{origin0}$
  - y: tank$_{origin1}$ - $T_{PlantWall}$
  - z: tank$_{origin2}$ + H_{Sed} - $H_{EChannel}$
- if layout3:
  - x: tank$_{origin0}$ - L_{Sed} + $W_{Channel}$ + $T_{ChannelWall}$ + $W_{EChannel}$
  - y: tank$_{origin1}$ - $T_{PlantWall}$
  - z: tank$_{origin2}$ + H_{Sed} - $H_{EChannel}$
- if layout4:
  - x: tank$_{origin0}$ - L_{Sed} + $W_{ChannelInlet}$ + $T_{ChannelWall}$ + $W_{EChannel}$

    - y: $tank_{origin1}$ - $T_{PlantWall}$
    - z: $tank_{origin2}$ + H_{Sed} - $H_{EChannel}$

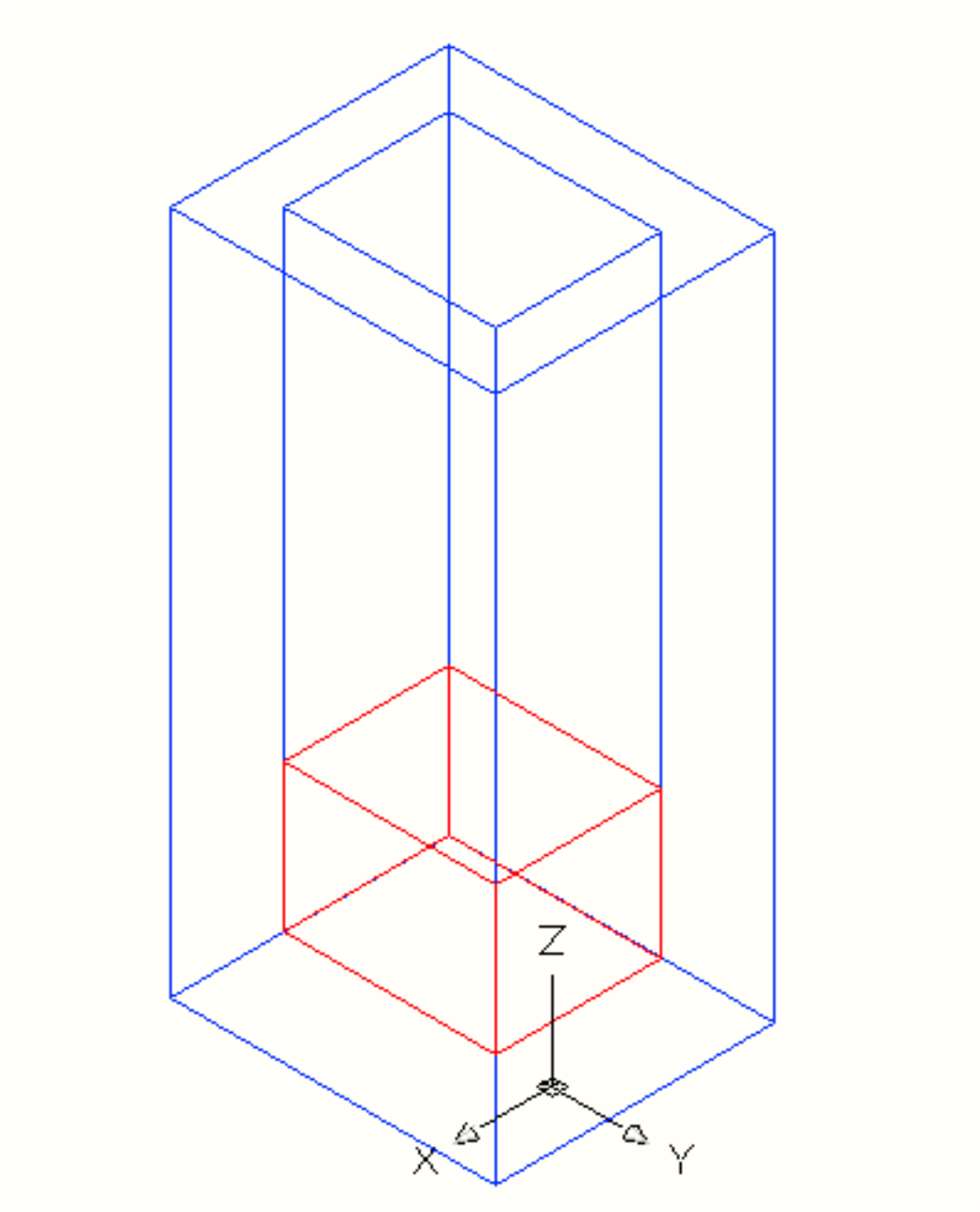
$echannelbox_{dim}$ =

- x: $W_{EChannel}$
- y: $N_{SedTanks(W}$Sed~ + $T_{PlantWall}$)
- z: $H_{EChannel}$

$T_{PlantWall}$ = Thickness of tank wall of sed tank and floc tank.



Top View



Top View

boxect - Creates a box based on two points.

boxect <-box($echanneltank_{origin}$,$echanneltank_{origin}$ + $echannelbox_{dim}$)

$echannel_{origin}$ =

- if layout1:
  - x: 0
  - y: 0
  - z: 0
- if layout2:
  - x: $tank_{origin0}$
  - y: $tank_{origin1}$ - $T_{PlantWall}$
  - z: $tank_{origin2}$ + H_{Sed} - $H_{EChannel}$
- if layout3:

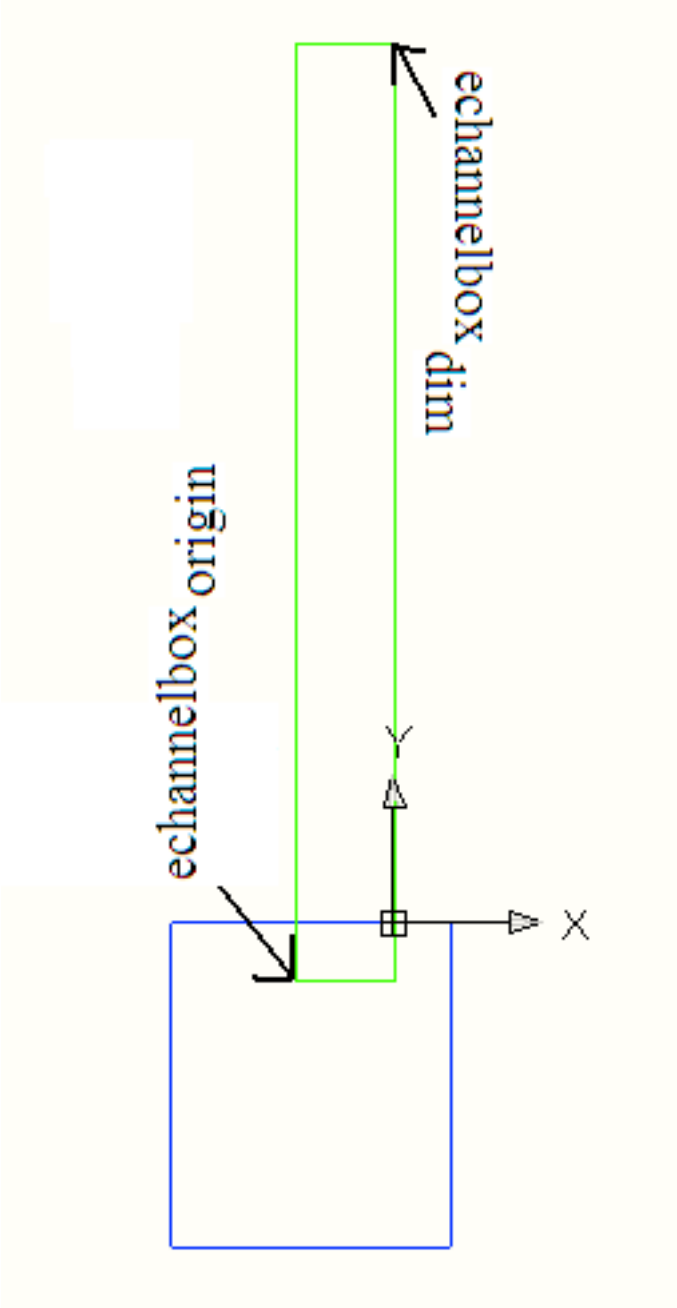
    - x: $tank_{origin0}$ - L_{Sed} + $W_{Channel}$ + $T_{ChannelWall}$ + $W_{EChannel}$
    - y: $tank_{origin1}$ - $T_{PlantWall}$
    - z: $tank_{origin2}$ + H_{Sed} - $H_{EChannel}$

- if layout4:
    - x: $tank_{origin0}$ - L_{Sed} + $W_{ChannelInlet}$ + $T_{ChannelWall}$ + $W_{EChannel}$
    - y: $tank_{origin1}$ - $T_{PlantWall}$

$echannelbox_{dim}$ =

- x: $W_{EChannel}$
- y: $N_{SedTanks(W}$Sed~ + $T_{PlantWall})$
- z: $H_{EChannel}$
  bigunion - $Union_{allA}$ selects all the objects visible in the workspace and joins them all into a single object.

bigunion <-$union_{allA}$



Top View

boxect - Creates a box based on two points.

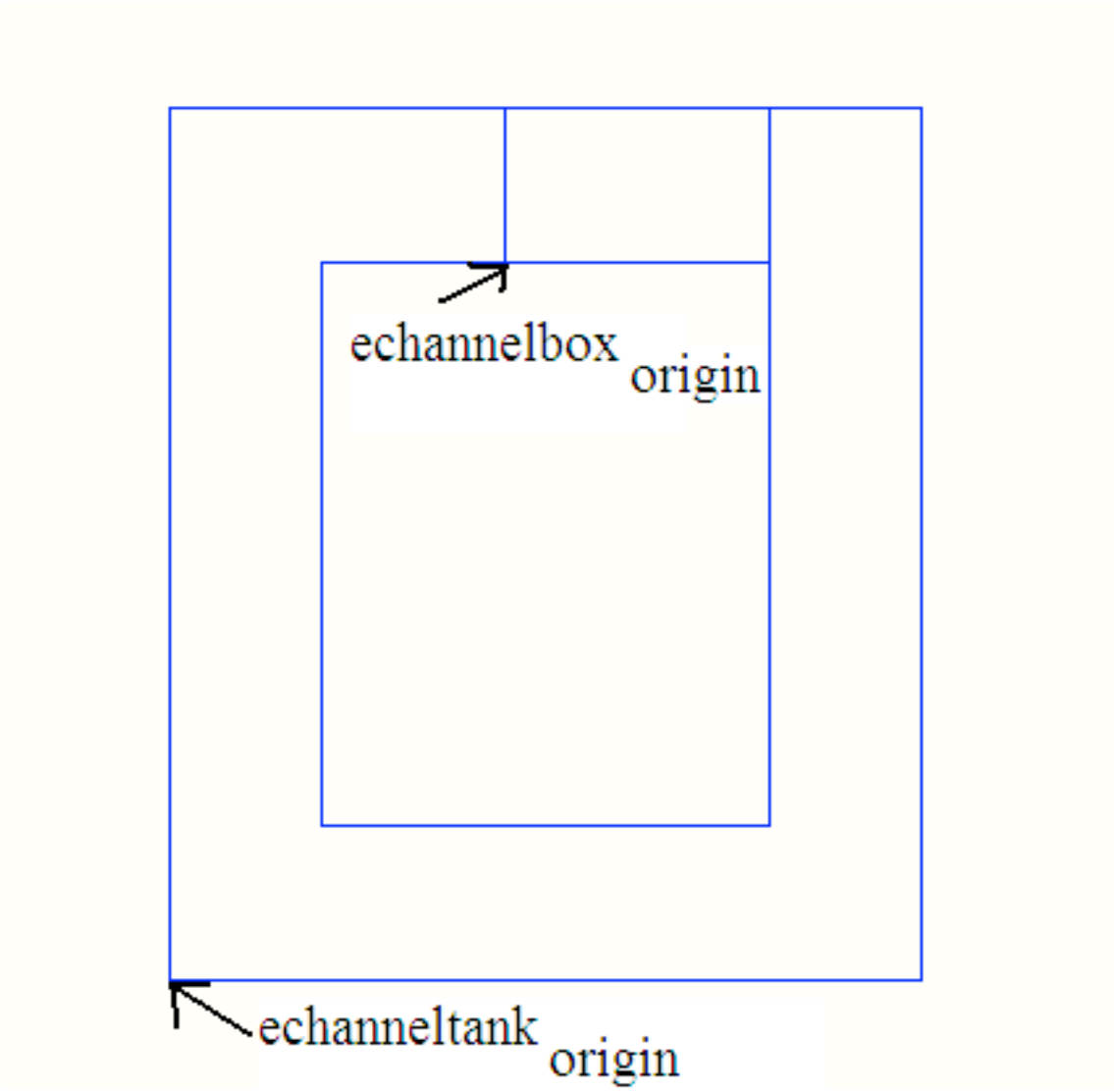
boxect <- box($echannelbox_{origin}$,$echannelbox_{origin}$ + $echannelbox_{dim}$)

$echannel_{origin}$ =

- if layout1:
  - x: 0
  - y: 0
  - z: 0
- if layout2:
  - x: $tank_{origin0}$
  - y: $tank_{origin1}$ - $T_{PlantWall}$
  - z: $tank_{origin2}$ + H_{Sed} - $H_{EChannel}$
- if layout3:
  - x: $tank_{origin0}$ - L_{Sed} + $W_{Channel}$ + $T_{ChannelWall}$ + $W_{EChannel}$
  - y: $tank_{origin1}$ - $T_{PlantWall}$
  - z: $tank_{origin2}$ + H_{Sed} - $H_{EChannel}$
- if layout4:
  - x: $tank_{origin0}$ - L_{Sed} + $W_{ChannelInlet}$ + $T_{ChannelWall}$ + $W_{EChannel}$
  - y: $tank_{origin1}$ - $T_{PlantWall}$

$echannelbox_{dim}$ =

- x: $W_{EChannel}$
- y: $N_{SedTanks(W}$Sed~ + $T_{PlantWall}$)
- z: $H_{EChannel}$



Top View

subtractect - SubtractD subtracts one object from the other based on two points.

subtractect <- subtractD(echanneltank$_{origin}$,echannelbox$_{origin}$)

layerset - Layer$_{set}$ selects the layer "0".

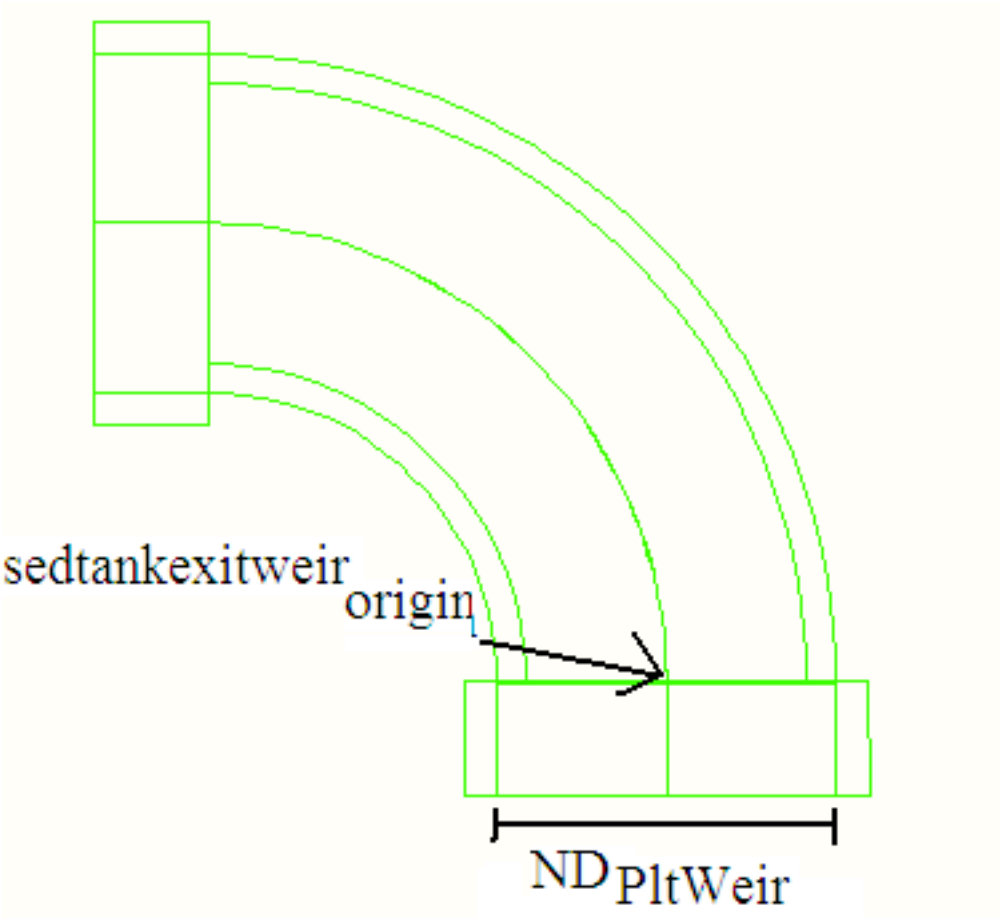
layerset <-layer$_{set}$("0")

layerfreezeect - Layer$_{freeze}$ locks the layer "echanneltank" so that it cannot be edited.

layerfreezeect <-layer$_{freeze}$("echanneltank")

layernewewe - Layer$_{new}$ creates a new green layer, "exitweirelbow."

layernewewe <- layer$_{new}$("exitweirelbow",green)

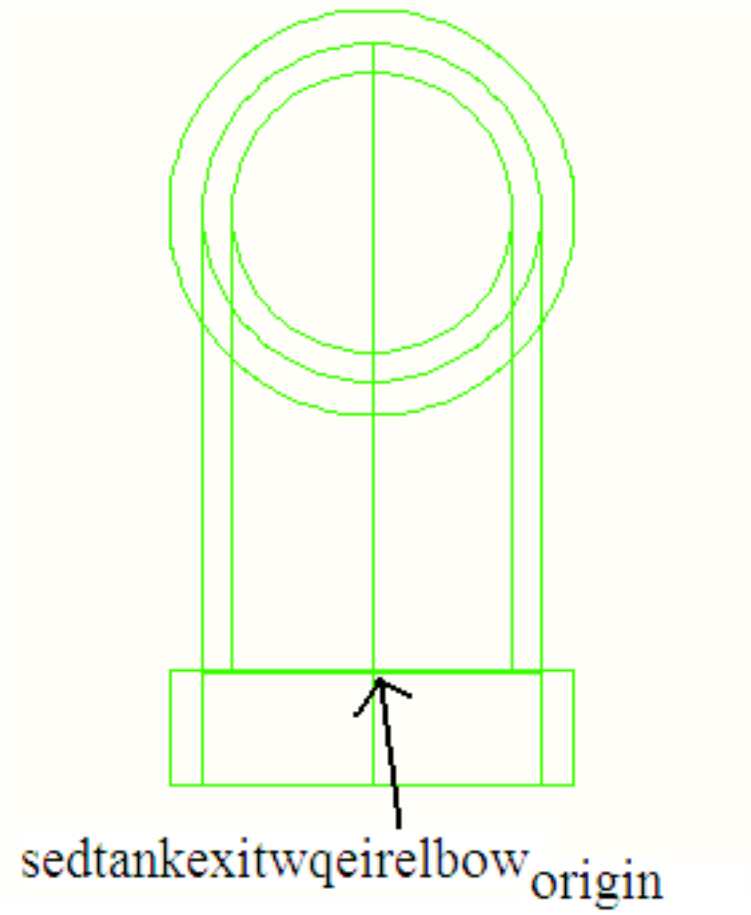


Top View

elbowewe - Calls the Elbow Program to draw an elbow.

elbowewe <-elbow(sedtankexitweirelbow$_{origin}$,ND$_{PltWeir}$,EN$_{PipeSpec}$)

EN$_{PipeSpec}$ - enumerated type



Top View

rotateewe1 - Rotate$_{3d}$ rotates the selected object by the designated degrees.

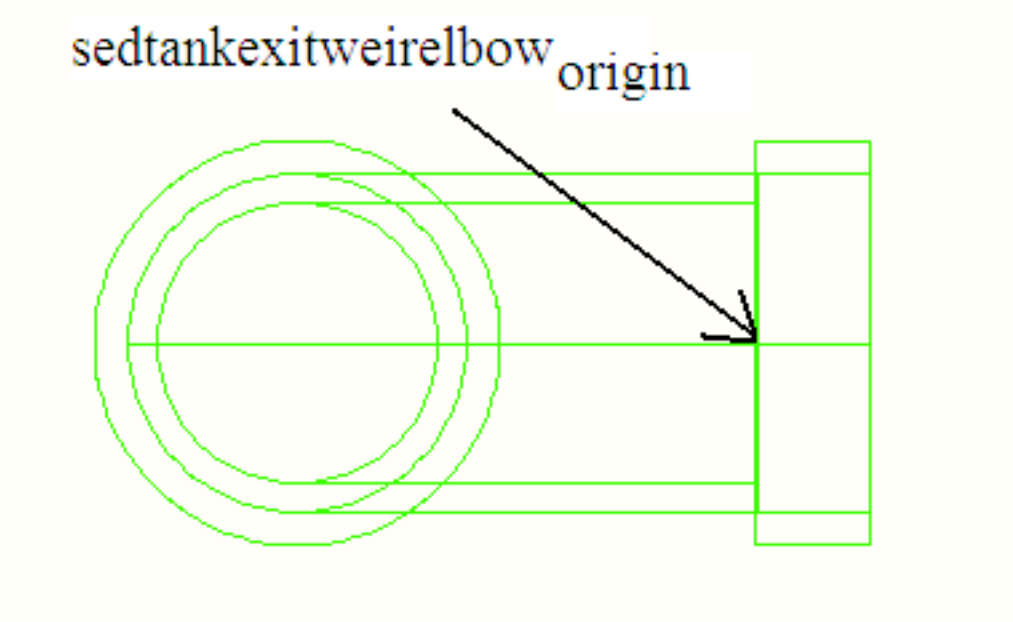
rotateewe1 <- rotate$_{3d}$(p1,sedtankexitweirelbow$_{origin}$,"y",90)

p1 =

- x: sedtankexitweirelbow$_{origin0}$ + ElbowRadius(ND$_{PltWeir}$)
- y: sedtankexitweirelbow$_{origin1}$
- z: sedtankexitweirelbow$_{origin2}$ - outerradius(ND$_{PltWeir}$)

"y" = rotation axis

90 = rotation angle



Top View

rotateewe2 - Rotate$_{3d}$ rotates the selected object by the designated degrees.

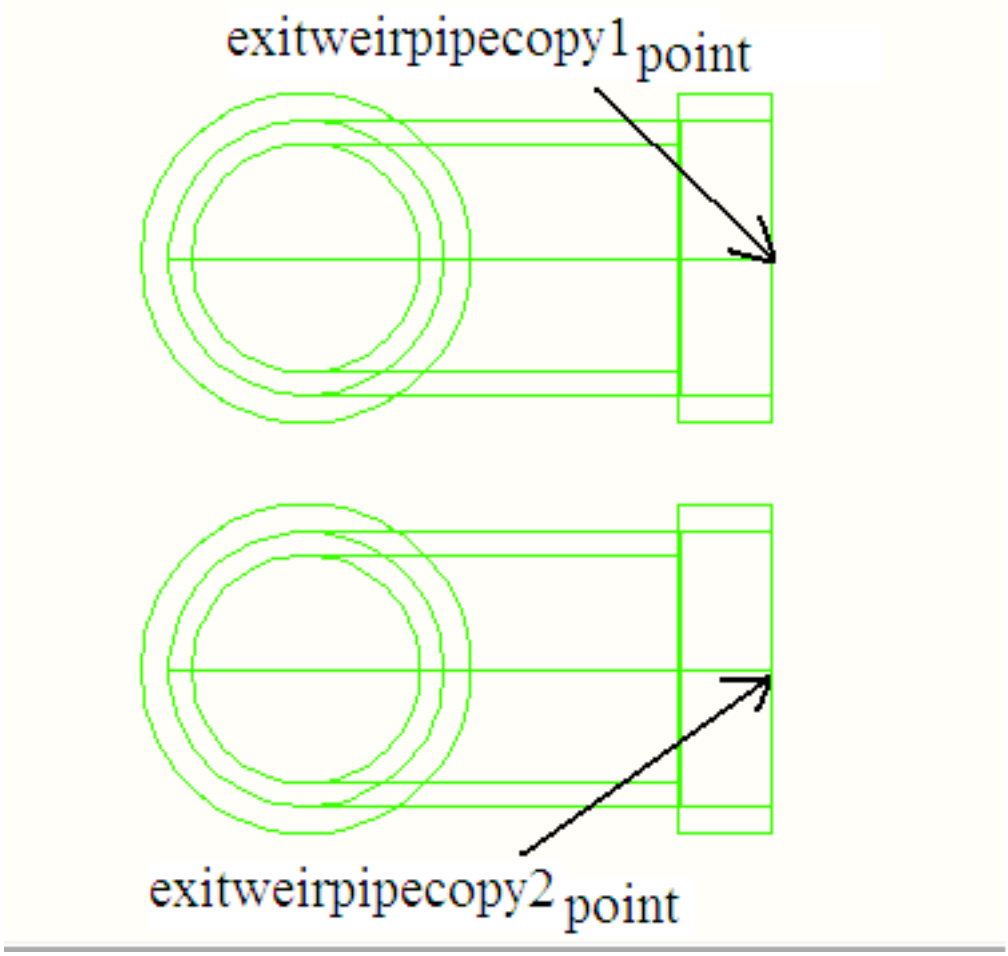
rotateewe2 <- rotate$_{3d}$(p1,sedtankexitweirelbow$_{origin}$,"z",90)

p1 =

- x: sedtankexitweirelbow$_{origin0}$
- y: sedtankexitweirelbow$_{origin1}$ + ElbowRadius(ND$_{PltWeir}$) - outerradius(ND$_{PltWeir}$)
- z: sedtankexitweirelbow$_{origin2}$

"z" = rotation axis

90 = rotation angle



Top View

copyewe - CopyB copies selected objects based on three selected points.

copyewe <- copyB(exitweirpipecopy1$_{point}$,exitweirpipecopy1$_{point}$,exitweirpipecopy2$_{point}$)

layerfreezeewp - Layer$_{freeze}$ locks the layer "exitweir" so that it cannot be edited.

layerfreezeewp <- layer$_{freeze}$("exitweir")

This page last changed on Dec 09, 2008 by ar329.

Sedimentation Tank Pipe Rotation Angles

$sedtankpipe5_{angle}$ = -(rad/deg) * atan((((5*L_{Sed})/8 - $W_{Channel}$/2)/ [3 * (W_{Sed}/2 - outerradius($ND_{SedLaunder}$))/4) + outerradius($ND_{SedLaunder}$))$^{-1}$)

$sedtankpipe6_{angle}$ = -(rad/deg) * ((((L_{Sed}/8 - $W_{Channel}$/2)/ ((W_{Sed}/2 - outerradius($ND_{SedLaunder}$))/4) + outerradius($ND_{SedLaunder}$))$^{-1}$)

$sedtankpipe7_{angle}$ = -(rad/deg) * atan(((3*L_{Sed}/8 - $W_{Channel}$/2)/ ((W_{Sed}/2 - outerradius($ND_{SedLaunder}$))/4) + outerradius($ND_{SedLaunder}$))$^{-1}$)

$sedtankpipe8_{angle}$ = -(rad/deg) * atan((((7*L_{Sed})/8 - $W_{Channel}$/2)/ (3*(W_{Sed}/2 - outerradius($ND_{SedLaunder}$))/4) + outerradius($ND_{SedLaunder}$))$^{-1}$)

This page last changed on Dec 18, 2008 by ar329.

Sedimentation Tank Pipe Lengths

$inletweirpipe_{length}$ = H_{Sed} - $H_{InletChannel}$ + $HW_{InletChannel}$ - ElbowRadius($ND_{PltWeir}$) - ConRadius($ND_{PltWeir}$)

$exitweirpipe_{length}$ = H_{Sed} - $H_{ExitChannel}$ + $H_{PltWeir}$ - ElbowRadius($ND_{PltWeir}$) - ConRadius(ND{~}PltWeir)

$sedtankpipe1_{length}$ = H_{Sed} - $H_{InletChannel}$ - (W_{Sed}/2 * tank($AN_{SedBottom}$) + ElbowRadius($ND_{SedPipeEntrance}$))-ElbowRadius($ND_{SedPipeEntrance}$)

$sedtankpipe5_{length}$ = $((5L_{Sed}/8 - W_{InletChannel}/2)^2$ + (outerradius($ND_{SedLaunder}$) + 3*(W_{Sed}/2 - outerradius($ND_{SedLaunder}$))/4)$^2)^{1/2}$ - 2ElbowRadius(ND$_{SedPipeEntrance\sim}$)

$sedtankpipe6_{length}$ = $((L_{Sed}/8 - W_{InletChannel}/2)^2$ + (outerradius($ND_{SedLaunder}$) + 1*(W_{Sed}/2 - outerradius($ND_{SedLaunder}$))/4)$^2)^{1/2}$ - 2ElbowRadius($ND_{SedPipeEntrance}$)

$sedtankpipe7_{length}$ = $((3L_{Sed}/8 - W_{InletChannel}/2)^2$ + (outerradius($ND_{SedLaunder}$) - 1*W_{Sed}/2 - outerradius($ND_{SedLaunder}$))/4)$^2)^{1/2}$ - 2ElbowRadius($ND_{SedPipeEntrance}$)

$sedtankpipe8_{length}$ = $((7L_{Sed}/8 - W_{InletChannel}/2)^2$ + (outerradius($ND_{SedLaunder}$) - 3*W_{Sed}/2 - outerradius($ND_{SedLaunder}$))/4)$^2)^{1/2}$ - 2ElbowRadius($ND_{SedPipeEntrance}$)

$sedtankpipe9_{length}$ = W_{Sed}/2 * tan($AN_{SedBottom}$) - ElbowRadius($ND_{SedPipeEntrance}$) - $S_{SedInlet}$ + ElbowRadius($ND_{SedPipeEntrance}$)

$sedtankpipe13_{length}$ = HW_{Sed} - $H_{SedAbove}$/2 - 2ElbowRadius($ND_{SedLaunder}$) - (H_{Sed} - $H_{InletChannel}$ - T_{Mp})

$sedtankpipe14_{length}$ = $W_{InletChannel}$ + $T_{ChannelWall}$

$manifold_{length}$ =

- if layout1 = L_{Sed} - $W_{InletChannel}$ - $T_{ChannelWall}$ - 2ElbowRadius($ND_{SedLaunder}$)
- if layout2 = L_{Sed} - $W_{InletChannel}$ - $T_{ChannelWall}$ - $W_{ExitChannel}$
- if layout3 = L_{Sed} - $W_{InletChannel}$ - $T_{ChannelWall}$ - $W_{ExitChannel}$
- if layout4 = L_{Sed} - $W_{InletChannel}$ - $T_{ChannelWall}$ - $W_{ExitChannel}$

$waste_{length}$ = L_{Sed}

$sedtankinletpipe_{length}$ = H_{Sed} - $H_{InletChannel}$ - outerdiameter($ND_{SedSludge}$)-$H_{SlopeRectangle}$

This page last changed on Dec 18, 2008 by ar329.

Sedimentation Tank Origin Points

$channel_{origin}$ =

- if layout 1-3:
  - x: $tank_{origin0}$ - L_{Sed} + $W_{Channel}$
  - y: $tank_{origin1}$ - $T_{PlantWall}$
  - z: $tank_{origin2}$ + H_{Sed} - $H_{Channel}$
- if layout4:
  - x: $tank_{origin0}$ - L_{Sed} -$T_{ChannelWall}$
  - y: $tank_{origin1}$ - $T_{PlantWall}$
  - z: $tank_{origin2}$ + H_{Sed} - $H_{Channel}$

$echannel_{origin}$ =

- if layout1:
  - x: 0
  - y: 0
  - z: 0
- if layout2:
  - x: $tank_{origin0}$
  - y: $tank_{origin1}$ - $T_{PlantWall}$
  - z: $tank_{origin2}$ + H_{Sed} - $H_{EChannel}$
- if layout3:
  - x: $tank_{origin0}$ - L_{Sed} + $W_{Channel}$ + $T_{ChannelWall}$ + $W_{EChannel}$
  - y: $tank_{origin1}$ - $T_{PlantWall}$
  - z: $tank_{origin2}$ + H_{Sed} - $H_{EChannel}$
- if layout4:
  - x: $tank_{origin0}$ - L_{Sed} + $W_{ChannelInlet}$ + $T_{ChannelWall}$ + $W_{EChannel}$
  - y: $tank_{origin1}$ - $T_{PlantWall}$
  - z: $tank_{origin2}$ + H_{Sed} - $H_{EChannel}$

$channelinletwall_{origin}$ =

- x: $tank_{origin0}$ - L_{Sed} + $W_{ChannelInlet}$
- y: $tank_{origin1}$
- z: $tank_{origin2}$

$channelinletwall_{dim}$ =

- x: $T_{ChannelWall}$
- y: W_{Sed}
- z: H_{Sed}

$lamina_{origin}$ =

- if layout1:
  - x: $tank_{origin0}$
  - y: $tank_{origin1}$
  - z: $tank_{origin2}$ + $H_{SedBelow}$
- if layout2:
  - x: $tank_{origin0}$ - $W_{EChannel}$

    - y: $tank_{origin1}$
    - z: $tank_{origin2} + H_{SedBelow}$

- if layout3:
    - x: $tank_{origin0}$
    - y: $tank_{origin1}$
    - z: $tank_{origin2} + H_{SedBelow}$

- if layout4:
    - x: $tank_{origin0}$
    - y: $tank_{origin1}$
    - z: $tank_{origin2} + H_{SedBelow}$

$sedtankpipe1_{origin}$ =

- x: $Plant_{origin0} - L_{Sed} + W_{InletChannel}/2$
- y: $Plant_{origin1} + W_{Sed}/2 + outerradius(ND_{SedLaunder}) + 3*(W_{Sed}/2 - outerradius(ND_{SedLaunder}))/4$
- z: $Plant_{origin2} + H_{Sed} - H_{InletChannel}$

$sedtankpipe2_{origin}$ =

- x: $Plant_{origin0} - L_{Sed} + W_{InletChannel}/2$
- y: $Plant_{origin1} + W_{Sed}/2 + outerradius(ND_{SedLaunder}) + 1*(W_{Sed}/2 - outerradius(ND_{SedLaunder}))/4$
- z: $Plant_{origin2} + H_{Sed} - H_{InletChannel}$

$sedtankpipe3_{origin}$ =

- x: $Plant_{origin0} - L_{Sed} + W_{inletChannel}/2$
- y: $Plant_{origin1} + W_{Sed}/2 - outerradius(ND_{SedLaunder}) - 1*(W_{Sed}/2 - outerradius(ND_{SedLaunder}))/4$
- z: $Plant_{origin2} + H_{Sed} - H_{InletChannel}$

$sedtankpipe4_{origin}$ =

- x: $Plant_{origin0} - L_{Sed} + W_{InletChannel}/2$
- y: $Plant_{origin1} + W_{Sed}/2 - outerradius(ND_{SedLaunder}) - 3*(W_{Sed}/2 - outerradius(ND_{SedLaunder}))/4$
- z: $Plant_{origin2} + H_{Sed} - H_{InletChannel}$

$sedtankpipe5_{origin}$ =

- x: $Plant_{origin0} - L_{Sed} + W_{InletChannel}/2 + ElbowRadius(ND_{SedPipeEntrance})$
- y: $Plant_{origin1} + W_{Sed}/2 + outerradius(ND_{SedLaunder}) + 3*(W_{Sed}/2 - outerradius(ND_{SedLaunder}))/4$
- z: $Plant_{origin2} + W_{Sed}/2*\tan(AN_{SedBottom}) + ElbowRadius(ND_{SedPipeEntrance})$

$sedtankpipe6_{origin}$ =

- x: $Plant_{origin0} - L_{Sed} + W_{InletChannel}/2 + ElbowRadius(ND_{SedPipeEntrance})$
- y: $Plant_{origin1} + W_{Sed}/2 + outerradius(ND_{SedLaunder}) + 1*(W_{Sed}/2 - outerradius(ND_{SedLaunder}))/4$
- z: $Plant_{origin2} + W_{Sed}/2*\tan(AN_{SedBottom}) + ElbowRadius(ND_{SedPipeEntrance})$

$sedtankpipe7_{origin}$ =

- x: $Plant_{origin0} - L_{Sed} + W_{InletChannel}/2 + ElbowRadius(ND_{SedPipeEntrance})$
- y: $Plant_{origin1} + W_{Sed}/2 - outerradius(ND_{SedLaunder}) - 1*(W_{Sed}/2 - outerradius(ND_{SedLaunder}))/4$
- z: $Plant_{origin2} + W_{Sed}/2*\tan(AN_{SedBottom}) + ElbowRadius(ND_{SedPipeEntrance})$

$sedtankpipe8_{origin}$ =

- x: $Plant_{origin0} - L_{Sed} + W_{InletChannel}/2 + ElbowRadius(ND_{SedPipeEntrance})$
- y: $Plant_{origin1} + W_{Sed}/2 - outerradius(ND_{SedLaunder}) - 3*(W_{Sed}/2 - outerradius(ND_{SedLaunder}))/4$
- z: $Plant_{origin2} + W_{Sed}/2*\tan(AN_{SedBottom}) + ElbowRadius(ND_{SedPipeEntrance})$

$sedtankpipe9_{origin}$ =

- x: $Plant_{origin0}$ - (7*L_{Sed})/8
- y: $Plant_{origin1}$ + W_{Sed}/2
- z: $Plant_{origin2}$ + $S_{SedInlet}$

$sedtankpipe10_{origin}$ =

- x: $Plant_{origin0}$ - (5*L_{Sed})/8
- y: $Plant_{origin1}$ + W_{Sed}/2
- z: $Plant_{origin2}$ + $S_{SedInlet}$

$sedtankpipe11_{origin}$ =

- x: $Plant_{origin0}$ - (3*L_{Sed})/8
- y: $Plant_{origin1}$ + W_{Sed}/2
- z: $Plant_{origin2}$ + $S_{SedInlet}$

$sedtankpipe12_{origin}$ =

- x: $Plant_{origin0}$ - (L_{Sed})/8
- y: $Plant_{origin1}$ + W_{Sed}/2
- z: $Plant_{origin2}$ + $S_{SedInlet}$

$sedtankpipe13_{origin}$ =

- x: $Plant_{origin0}$ - L_{Sed} + $W_{InletChannel}$ + $T_{ChannelWall}$ + ElbowRadius($ND_{SedLaunder}$)
- y: $Plant_{origin1}$ + W_{Sed}/2
- z: $Plant_{origin2}$ + H_{Sed} - $H_{InletChannel}$ - T_{Mp} + ElbowRadius($ND_{SedLaunder}$)

$sedtankpipe14_{origin}$ =

- x: $Plant_{origin0}$ - L_{Sed}
- y: $Plant_{origin1}$ + W_{Sed}/2
- z: $Plant_{origin2}$ + H_{Sed} - $HInlet_{Channel}$ - T_{Mp}

$sedtankbox1_{origin}$ =

- x: $Plant_{origin0}$ - L_{Sed}
- y: $Plant_{origin1}$ + W_{Sed}/2 - outerradius($ND_{SedSludge}$)
- z: $Plant_{origin2}$

$sedtankbox1_{dim}$ =

- x: L_{Sed}
- y: outerdiameter($ND_{SedSludge}$)
- z: 3*outerdiameter($ND_{SedSludge}$)

$manifold_{origin}$ =

- if layout 1:
  - x: $tank_{origin0}$ - L_{Sed} + $W_{Channel}$ + $T_{ChannelWall}$ + 2*ElbowRadius($ND_{SedLaunder}$)
  - y: $tank_{origin1}$ + W_{Sed}/2
  - z: $tank_{origin2}$ + HW_{Sed} - $H_{SedAbove}$/2
- if layout 2:
  - x: $tank_{origin0}$ - L_{Sed} + $W_{Channel}$ + $T_{ChannelWall}$
  - y: $tank_{origin1}$ + W_{Sed}/3
  - z: $tank_{origin2}$ + HW_{Sed} - $H_{SedAbove}$/2
- if layout 3:
  - x: $tank_{origin0}$ - L_{Sed} + $W_{Channel}$ + $T_{ChannelWall}$ + $W_{EChannel}$
  - y: $tank_{origin1}$ + W_{Sed}/3
  - z: $tank_{origin2}$ + HW_{Sed} - $H_{SedAbove}$/2

- if layout 4:
  - x: $tank_{origin0}$ - L_{Sed} + $W_{ChannelInlet}$ + $T_{ChannelWall}$ + $W_{EChannel}$
  - y: $tank_{origin1}$ + W_{Sed}/3
  - z: $tank_{origin2}$ + HW_{Sed} - $H_{SedAbove}$/2

$manifold2_{origin}$ =

- if layout 1:
  - x: 0
  - y: 0
  - z: 0
- if layout 2:
  - x: $tank_{origin0}$ - L_{Sed} + $W_{Channel}$ + $T_{ChannelWall}$
  - y: $tank_{origin1}$ + 2*W_{Sed}/3
  - z: $tank_{origin2}$ + HW_{Sed} - $H_{SedAbove}$/2
- if layout 3:
  - x: $tank_{origin0}$ - L_{Sed} + $W_{Channel}$ + $T_{ChannelWall}$ + $W_{EChannel}$
  - y: $tank_{origin1}$ + 2*W_{Sed}/3
  - z: $tank_{origin2}$ + HW_{Sed} - $H_{SedAbove}$/2
- if layout 4:
  - x: $tank_{origin0}$ - L_{Sed} + $W_{ChannelInlet}$ + $T_{ChannelWall}$ + $W_{EChannel}$
  - y: $tank_{origin1}$ + 2*W_{Sed}/3
  - z: $tank_{origin2}$ + HW_{Sed} - $H_{SedAbove}$/2

$waste_{origin}$

- x: $tank_{origin0}$ - L_{Sed}
- y: $tank_{origin1}$ + W_{Sed}/2
- z: $tank_{origin2}$ + outerdiameter($ND_{SedSludge}$)/2

$channelbox_{origin}$ =

- x: $tank_{origin0}$ - L_{Sed}
- y: $tank_{origin1}$
- z: $tank_{origin2}$ + H_{Sed} - $H_{Channel}$

$echannelbox_{origin}$ =

- if layout 1:
  - x: 0
  - y: 0
  - z: 0
- if layout 2:
  - x: $tank_{origin0}$ - $W_{EChannel}$
  - y: $tank_{origin1}$ - $T_{PlantWall}$
  - z: $tank_{origin2}$ + H_{Sed} - $H_{EChannel}$
- if layout 3:
  - x: $tank_{origin0}$ - L_{Sed} + $W_{Channel}$ + $T_{ChannelWall}$
  - y: $tank_{origin1}$ - $T_{PlantWall}$
  - z: $tank_{origin2}$ + H_{Sed} - $H_{EChannel}$
- if layout 4:
  - x: $tank_{origin0}$ - L_{Sed} + $W_{ChannelInlet}$ + $T_{ChannelWall}$
  - y: $tank_{origin1}$ - $T_{PlantWall}$
  - z: $tank_{origin2}$ + H_{Sed} - $H_{EChannel}$

$channelinletbox1_{origin}$ =

- x: $tank_{origin0}$ - L_{Sed} + $T_{PlantWall}$
- y: $tank_{origin1}$ + .5*W_{Sed} - .5*$L_{ChannelInlet}$
- z: $tank_{origin2}$ + H_{Sed} - $H_{Channel}$

$channelinletbox2_{origin}$ =

- x: $tank_{origin0}$ - L_{Sed}
- y: $tank_{origin1}$
- z: $tank_{origin2}$

$channelclosebox_{origin}$ =

- x: $tank_{origin0}$ - L_{Sed} - $T_{PlantWall}$ - $W_{Channel}$ - $T_{ChannelWall}$
- y: $tank_{origin1}$ - $T_{PlantWall}$
- z: $tank_{origin2}$ + H_{Sed} - $H_{Channel}$ - $T_{ChannelWall}$

$channelbox_{dim}$ =

- x: $W_{Channel}$
- y: $N_{SedTanks(W}$Sed~ + $T_{PlantWall}$)
- z: $H_{Channel}$

$echannelbox_{dim}$ =

- x: $W_{EChannel}$
- y: $N_{SedTanks(W}$Sed~ + $T_{PlantWall}$)
- z: $H_{EChannel}$

$channelinletbox1_{dim}$ =

- x: $T_{PlantWall}$
- y: $L_{ChannelInlet}$
- z: $H_{Channel}$

$channelinletbox2_{dim}$ =

- x: $W_{ChannelInlet}$
- y: W_{Sed}
- z: H_{Sed}

$channelclosebox_{dim}$ =

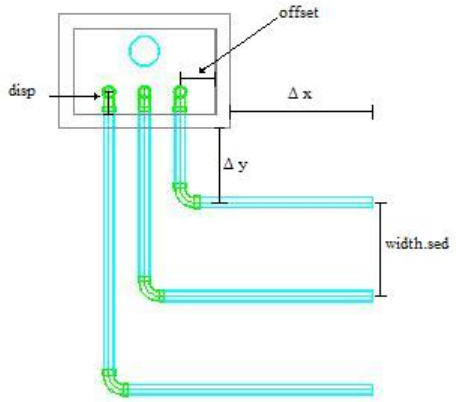
- x: $W_{Channel}$ + $T_{ChannelWall}$ + T{~}PlantWall
- y: T{~}PlantWall
- z: $H_{Channel}$ + $T_{ChannelWall}$

$inletslopes_{origin}$ =

- x: $tank_{origin0}$ - L_{Sed}
- y: $tank_{origin1}$ + W_{Sed}/2
- z: $tank_{origin2}$

This page last changed on Oct 04, 2008 by ar329.

General Program Information



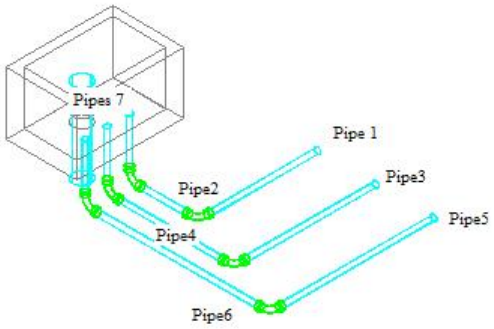
Input Definitions

- **origin** is a 3,1 matrix with the x,y,z values of the center point of the outlet of the first sedimentation tank
- **width** $_{sed}$ is the distance between the outlets of the sedimentation tanks or the width of the sedimentation tanks from the center line of the wall on either side of a tank
- **lvltank** $_{dim}$ is a 3,1 matrix with the x,y,z dimensions of the tank
- **# x** is the distance in the x direction in a top view from the outlet of the first sedimentation tank (or the origin of the first pipe) to the outside wall of the leveling tank
- **# y** is the distance in the y direction in a top view from the outlet of the first sedimentation tank (or the origin of the first pipe) to the outside wall of the leveling tank
- **# z** is the distance in the z direction in a top view from the outlet of the first sedimentation tank (or the origin of the first pipe) to the bottom wall of the platform
- **lvltank** $_{thick}$ is the thickness of the walls of the leveling tank
- **thick** $_{platform}$ is the thickness of the platform
- **R1** is the inner radius of the pipes leading to the leveling tank
- **R** $_{main}$ is the inner radius of the effluent pipe
- **H** $_{main}$ is the height of the effluent pipe (or the water level in the leveling tank)
- **n** is the number of sedimentation tanks
- **disp** is the displacement of the center of the pipes coming up into the leveling tanks from the inside wall of the tank

Notes

- All coordinates correspond to a top view.
- The floor of the leveling tank is at the same elevation as the top of the platform

Pipe Labeling



Step by Step Description

The Sed $_{eff}$ program is composed of three scripts. The program will run Sedeff $_{script1}$ when n is 3, Sedeff $_{script2}$ when n is 4 and Sedeff $_{script3}$ when n is 5. Because the most common number of sedimentation tanks is three, the output of Sedeff $_{script1}$ will be described below.

Some points and values which will be used several times in the script are defined outside of the script so as to make it cleaner. These variables will be pseudo inputs for Sedeff $_{script1}$. They are listed below and will be described in greater detail further down.

- pipe2 $_{origin}$, pipe3 $_{origin}$, pipe4 $_{origin}$, pipe5 $_{origin}$, pipe6 $_{origin}$, elbow1 $_{origin}$, pipe7 $_{origin}$, tank $_{origin}$, main $_{origin}$, origin, lvltank $_{dim}$, #x, #y, #z, tank $_{thick}$, thick $_{platform}$, width $_{sed}$, R1, R $_{main}$, H $_{main}$, n, disp

Pseudo Inputs

pipe3$_{origin}$

pipe4$_{origin}$

pipe5$_{origin}$

pipe6$_{origin}$

elbow1$_{origin}$

pipe7$_{origin}$

tank$_{origin}$

main$_{origin}$

origin

lvltank$_{dim}$

#x

#y

#z

thick$_{tank}$

thick$_{platform}$

$width_{sed}$

R1

R_{main}

H_{main}

n - number of sedimentation tanks to be drawn by the program.

disp

layer1 - Creates a 'lvlpipe' layer for all of the pipes in the program

Pipe1 - Creates a pipe with a radius of R1 and an origin at the variable "origin". See Pipe Help for details on the pipe function.

Pipe1 Length = #x + thick $_{tank}$ + offset - ElbowRadius
Notes:

- Pipe1 Length is not a variable, the values are inputed directly into the pipe function
- offset is a function which divides the x dimension of the level tank by the number of sedimentation tanks, so as to evenly space the inlet pipes to the tank.
- ElbowRadius is the radius of curvature of an elbow. In the script it is given as a function of the inner radius. This function is defined in the [Pipe Database program|]

Rotate1 - Pipe 1 is rotated using the Rotate3d function so that from a top view it is oriented in the west to east direction

p1:

- x: $origin_0$ + R1
- y: $origin_1$
- z: $origin_2$

p2: origin

axis: "y"

#: -90

Pipe2 - Creates a pipe with a radius of R1 and an origin at the variable "$pipe2_{origin}$".

pipe2 $_{origin}$ =

- x: x,origin - Pipe1 Length
- y: y,origin + ElbowRadius
- z: z,origin

Pipe2 Length = #y - 2ElbowRadius + thick $_{tank}$ + disp

Rotate2 - Pipe 2 is rotated using the Rotate3d function so it is oriented north to south in top view

p1:

- x: $pipe2_{origin0}$ + R1
- y: $pipe2_{origin1}$
- z: $pipe2_{origin2}$

p2: $pipe2_{origin}$

axis: "x"

#: -90

Pipe3 -Creates a pipe with a radius of R1 and an origin at the variable "$pipe3_{origin}$".

Pipe3 Length = #x + $thick_{tank}$ + 2offset($lvltank_{dim0}$,n) - elbowRadius(2R1)

Rotate3 Pipe 3 is rotated using the Rotate3d function so that from a top view it is oriented in the west to east direction

p1:

- x: $pipe3_{origin0}$ + R1
- y: $pipe3_{origin1}$
- z: $pipe3_{origin2}$

p2: $pipe3_{origin}$

axis: "y"

#: -90

Pipe4 -Creates a pipe with a radius of R1 and an origin at the variable "$pipe4_{origin}$".

Pipe4 Length = #y + $width_{sed}$ - 2ElbowRadius(2R1) + $thick_{tank}$ + disp

Rotate4 Pipe 4 is rotated using the Rotate3d function so that from a top view it is oriented in the west to east direction.

p1:

- x: $pipe4_{origin0}$ + R1
- y: $pipe4_{origin1}$
- z: $pipe4_{origin2}$

p2: $pipe4_{origin}$

axis: "y"

#: -90

Pipe5 -Creates a pipe with a radius of R1 and an origin at the variable "$pipe5_{origin}$".

Pipe5 length: #x + $thick_{tank}$ + 3offset($lvltank_{dim0}$,n) - ElbowRadius(2R1)

Rotate5 Pipe 5 is rotated using the Rotate3d function so that from a top view it is oriented in the west to east direction.

P1:

- x: $pipe5_{origin0}$ + R1
- y: $pipe5_{origin1}$
- z: $pipe5_{origin2}$

p2: $pipe5_{origin}$

axis: "y"

#: -90

Pipe6 -Creates a pipe with a radius of R1 and an origin at the variable "$pipe6_{origin}$".

Pipe6 length: #y + 2$wideth_{sed}$ - 2ElbowRadius(2R1) + $thick_{tank}$ + disp

Rotate6 - Pipe 6 is rotated using the Rotate3d function so it is oriented north to south in top view

Pipe7 -Creates a pipe with a radius of R1 and an origin at the variable "$pipe7_{origin}$".

Pipe7 length: #z - ElbowRadius(2R1) + $thick_{platform}$

zoom$_{win1}$ - Zoom$_{win}$creates a close-up window by using two points to specify the window size the object is to be viewed in.

p1: tank$_{origin}$

p2:

- x: tank$_{origin0}$ - lvltank$_{dim0}$
- y: tank$_{origin1}$ + lvltank$_{dim1}$
- z: tank$_{origin2}$

Copy5 - CopyCcreates a copy of the selected objects based on four chosen points.

p1:

- x: pipe7$_{origin0}$ + R1
- y: pipe7$_{origin1}$
- z: pipe7$_{origin2}$

p2: pipe7$_{origin}$

p3:

- x: -offset(lvltank$_{dim0}$,n)
- y: 0
- z: 0

p4:

- x: -2offset(lvltank$_{dim0}$,n)
- y: 0
- z: 0

Pipe8 -Creates a pipe with a radius of R$_{main}$ and an origin at the variable "main$_{origin}$".

Pipe8 length: H$_{main}$ + thick$_{platform}$ + #z

layer2 - Layer$_{set}$ select the layer "0"

layer3 - Layer$_{freeze}$ is used to freeze the layer "lvlpipe."

layer4 - Layer$_{new}$ is used to create a new, green layer "lvlelbow."

Elbow1 - Calls the Elbow Program program to build an elbow with the origin elbow1$_{origin}$ and an inner radius, R1.

Rotate7 - Pipe 7 is rotated using the Rotate3d function so...

p1:

- x: elbow1$_{origin0}$
- y: elbow1$_{origin1}$ + ElbowRadius(2R1)
- z: elbow1$_{origin2}$

p2:

- x: elbow1$_{origin0}$
- y: elbow1$_{origin1}$ + ElbowRadius(2R1)
- z: elbow1$_{origin2}$

axis: "z"

#: -180

zoom$_{win2}$

Copy1 - CopyBcreates a copy of the selected objects based on three chosen points.

p1: $pipe2_{origin}$

p2: $pipe2_{origin}$

p3: $pipe4_{origin}$

Copy2 - CopyBcreates a copy of the selected objects based on three chosen points.

p1: $pipe2_{origin}$

p2: $pipe2_{origin}$

p3: $pipe6_{origin}$

Copy3 - CopyBcreates a copy of the selected objects based on three chosen points.

p1: $pipe2_{origin}$

p2: $pipe2_{origin}$

p3: pipe11

Rotate8 - Pipe 2 is rotated using the Rotate3d function so it is oriented north to south in top view

p1:

- x: $pipe2_{origin0}$
- y: $pipe2_{origin1}$ + (#y - 2ElbowRadius(2R1) + $thick_{tank}$ + disp)
- z: $pipe2_{origin2}$

p2:

- x: $pipe2_{origin0}$
- y: $pipe2_{origin1}$ + (#y - 2ElbowRadius(2R1) + $thick_{tank}$ + disp)
- z: $pipe2_{origin2}$

axis: "x"

#: \180

Rotate9 - Pipe 2 is rotated using the Rotate3d function so that from a top view it is oriented in the west to east direction.

p1:

- x: $pipe2_{origin0}$
- y: $pipe2_{origin1}$ + (#y - 2ElbowRadius(2R1) + $thick_{tank}$ + disp
- z: $pipe2_{origin2}$

p2:

- x: $pipe2_{origin0}$
- y: $pipe2_{origin1}$ + (#y - 2ElbowRadius(2R1) + $thick_{tank}$ + disp
- z: $pipe2_{origin2}$

axis: "y"

#: -90

Copy4 CopyCcreates a copy of the selected objects based on four chosen points.

p1:

- x: $pipe2_{origin0}$

- y: $pipe2_{origin1}$ + (#y - 2ElbowRadius(2R1) + $thick_{tank}$ + disp
- z: $pipe2_{origin2}$

p2:

- x: $pipe2_{origin0}$
- y: $pipe2_{origin1}$ + (#y - 2ElbowRadius(2R1) + $thick_{tank}$ + disp
- z: $pipe2_{origin2}$

p3:

- x: -offset($lvltank_{dim0}$,n)
- y: 0
- z: 0

p4:

- x: -2offset($lvltank_{dim0}$,n)
- y: 0
- z: 0

layer5 - $Layer_{set}$ select the layer "0"

layer6 - $Layer_{freeze}$ freezes the layer "lvlelbow."

layer7 - $Layer_{new}$ creates a new grey layer "lvltank."

Tank - Calls the Tank Program program to build a tank with the origin $tank_{origin}$, tank dimensions $tank_{dim}$ and thickness $thick_{tank}$.

$tank_{origin}$ - a 3 by 1 matrix with x,y,z positions corresponding to the point where the tank will be drawn

$tank_{dim}$ - a 3 by 1 matrix with x,y,z positions corresponding to the length, width and height dimensions of the tank

- x: $tank.dim_0$ = length
- y: $tank.dim_1$ = width
- z: $tank.dim_2$ = height

thick - the thickness of the wall of the tank

layer8 - $Layer_{thaw}$ unfreezes the layer "lvlpipe."

layer9 - $Layer_{thaw}$ unfreezes the layer "lvlelbow."

layer10 - $Layer_{set}$ select the layer "lvlpipe."

cylinder - CylinderCcreates a cylinder based on a point, radius and length.

p1: $main_{origin}$

R1: R_{main}

L: H_{main} + $thick_{platform}$ + #z

subtract -SubtractD subtracts based on two points.

p1: $tank_{origin}$

p2:

- x: $main_{origin0}$ + R_{main}
- y: $main_{origin1}$
- z: $main_{origin2}$

one

two

three

return

AguaClara : AutoCAD Lamina Program

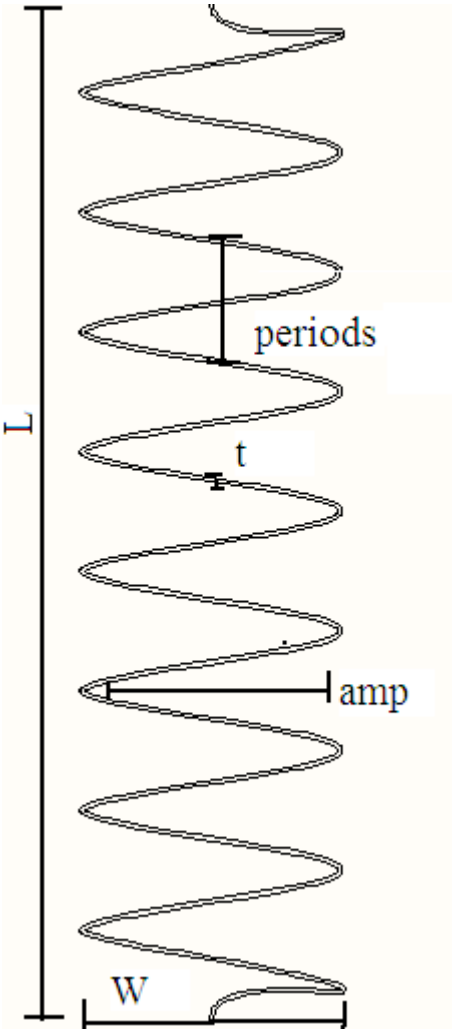
This page last changed on Dec 18, 2008 by ar329.

General Program Information

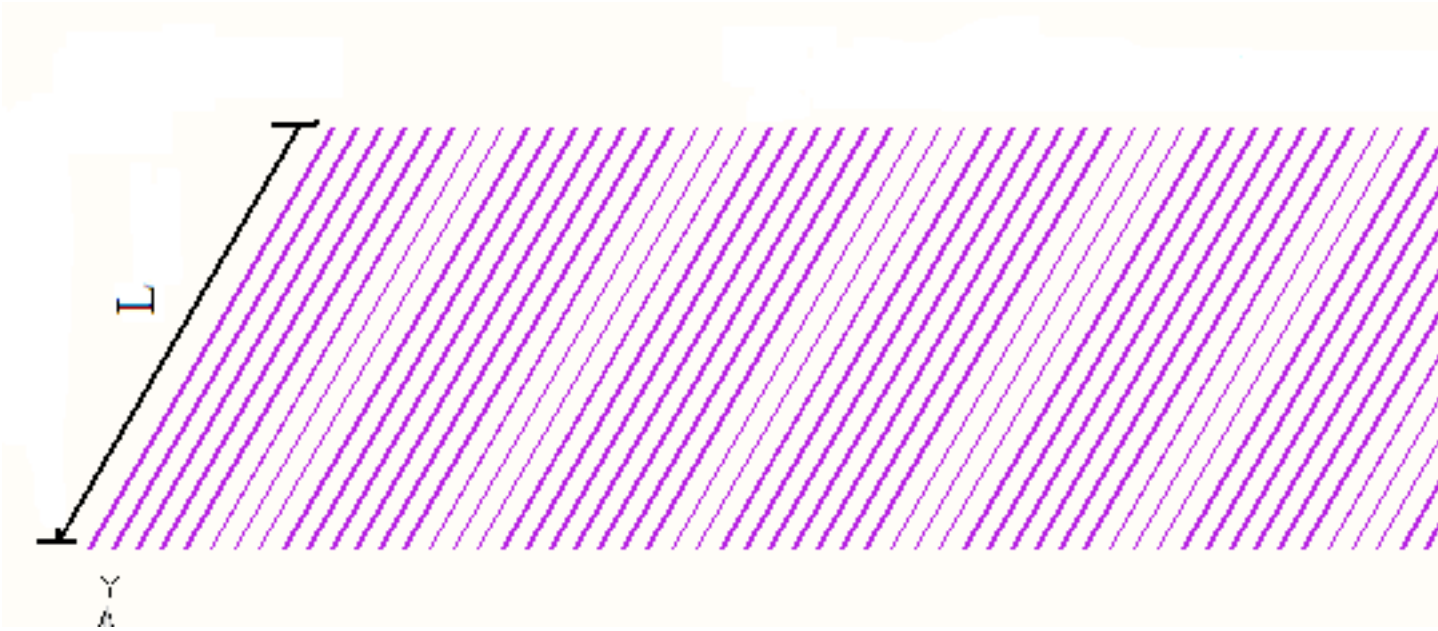
Input Definitions

Inputs Needed to Call the Channel Function

Inputs Defined within the Channel Function



Top View



Front View

origin - is a 3*1 matrix with x,y,z positions corresponding to the point where the first lamella will be drawn.

amp - specifies the amplitude of the lamina

w - specifies the width of an individual lamina

l - specifies the length of an individual lamina

t - specifies the thickness of an individual lamina

periods - specifies the period of one curve of the lamina.

angle - specifies the angle of the lamina relative to the floor of the tank.

$num_{baffles}$ - specifies the number baffles in the tank

$num_{bafflescol}$ -

xspace -

yspace -

Technical Program Outline

$lamina_0$ - The $zoom_{winA}$ function creates a close-up view based on two points to specify the window size.

p1:

- x: $origin_0$ - w - zc
- y: $origin_1$
- z: $origin_2$

p2:

- x: $origin_0$ + w + zc
- y: $origin_1$ + 1 + zc
- z: $origin_2$

for loop - a for loop is created for the range from i = 0 to i = (periods*4).

$lamina_{last}$ - Every time the program runs through the for loop another sine wave is drawn.

$lamina_{last}$:

- x: $origin_0$ + sin(#i/2)*amp
- y: $origin_1$ + i*(x/(periods*4)
- z: $origin_2$

$lamina_{i+1}$ - Uses the 'point' function to turn the 3*1 matrix $lamina_{last}$ into a text format that AutoCad can read.

$lamina_{i+2}$ - Uses the 'concat' function create a character string out of a space and the point (origin - zc_{point}).

$lamina_{i+3}$ - Uses the 'point' function to turn the 3*1 matrix ($lamina_{last}$ + zc_{point}) into a text format that AutoCad can read.

$lamina_{i+4}$ - Enters the command "offset" into AutoCad. The offset function is used to move the selected object a specified distance from its original location.

$lamina_{i+5}$ - The command 'stringit' turns the the variable "t" into a dimensionless number and cuts the number off after 5 decimal points. t represents the distance that the selected object is to be offset.

$lamina_{i+6}$ - Uses the 'point' function to turn the 3*1 matrix 'origin' into a text format that AutoCad can read.

$lamina_{i+7}$ - Uses the 'point' function to turn the following 3*1 matrix into a text format that AutoCad can read.

- x: $origin_0$ + t
- y: $origin_1$
- z: $origin_2$

$line_0$ - Uses the 'concat' function to create a character string out of the point 'origin' a space and the point converted in line $lamina_{i+7}$.

$line_1$ - Uses the 'concat' function to creat a character string out of the point '$lamina_{last}$' a space and the point described below into Autocad.

- x: $lamina_{last0}$ + t
- y: $lamina_{last1}$
- z: $lamina_{last2}$

$lamina_{i+8}$ - Uses the 'concat' function to create a character string out of a space, the AutoCad command "line," a space, the character string created in $line_0$, and a space. The purpose of the character string is to create a line in AutoCad between the two points specified in $line_0$.

$lamina_{i+9}$ - Uses the 'concat' function to create a character string out of a the AutoCad command "line," a space, the character string created in $line_1$ and a space. The purpose of the character string is to create a line in AutoCad between the two points specified in $line_1$.

$lamina_{i+10}$ - Uses the 'concat' function to create a character string out of the AutoCad command "z e region w," a space, a point described below, a space, the second point described below and a space. The purpose of the character string is to create a region based on the two points specified.

first point:

- x: $origin_0$ - amp
- y: $origin_1$
- z: $origin_2$

second point:

- x: $lamina_{last0}$ + t + amp
- y: $lamina_{last1}$

- z: $lamina_{last2}$

z: zoom
e: extents
region: creates a region based on two specified points.
w: specifies corners for the window in which the region is to be created.

$lamina_{i+11}$ - Uses the concat function to create a character string out of the AutoCad command extrude, a space, the point 'origin,' a space, a space, the variable 'l,' a space and the "0."

$lamina_{i+12}$ - Uses the concat function to create a character string out of the AutoCad command rotate3d, a space, the point 'origin,' a space, a space, the command "y," a space and a num2str (-(90-angle/deg)) if the angle is not 90 degrees.

$lamina_{i+13}$ - Uses the concat functino to create a character string out of the AutoCad command array, a space, the point 'origin,' a space, a space, the command "R," a space, num2str($num_{bafflecols}$, a space,$num2str_{baffles}$ and a space.

$lamina_{i+14}$ - Together $lamina_{i+14}$ and $lamina_{i+15}$ for an if statement. $Lamina_{i+14}$ uses the concat function to create a character string that enters the dimensionless number "yspace" if $num_{bafflecols}$ is less that 1.

$lamina_{i+15}$ - Enters "" into the command line if the condition in $lamina_{i+14}$ is not forfilled.

AguaClara : AutoCAD Flocculation Tank Program - Floctankscript1

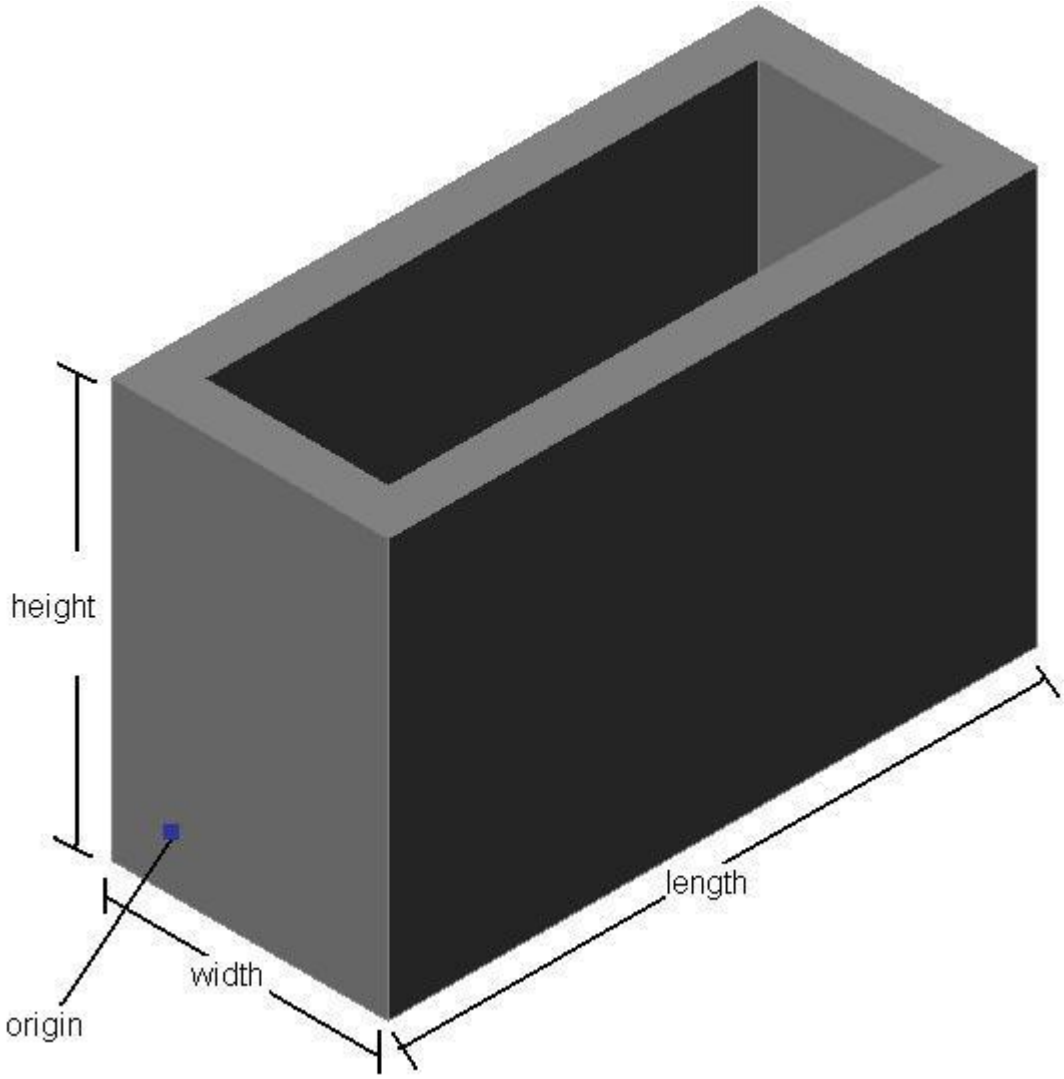
This page last changed on Nov 20, 2008 by ar329.

General Program Information

Technical Program Outline

layer1 - $Layer_{new}$ creates a new dark grey layer, "floctank."

layer1 <- $layer_{new}$("floctank",dkgrey)



Northeast Isometric View

tank1 - Calls the Tank Program to create a tank.

tank1 <- Tank($floc_{origin}$,$totalfloc_{dim}$,$tank_{thick}$)

$floc_{origin}$ =

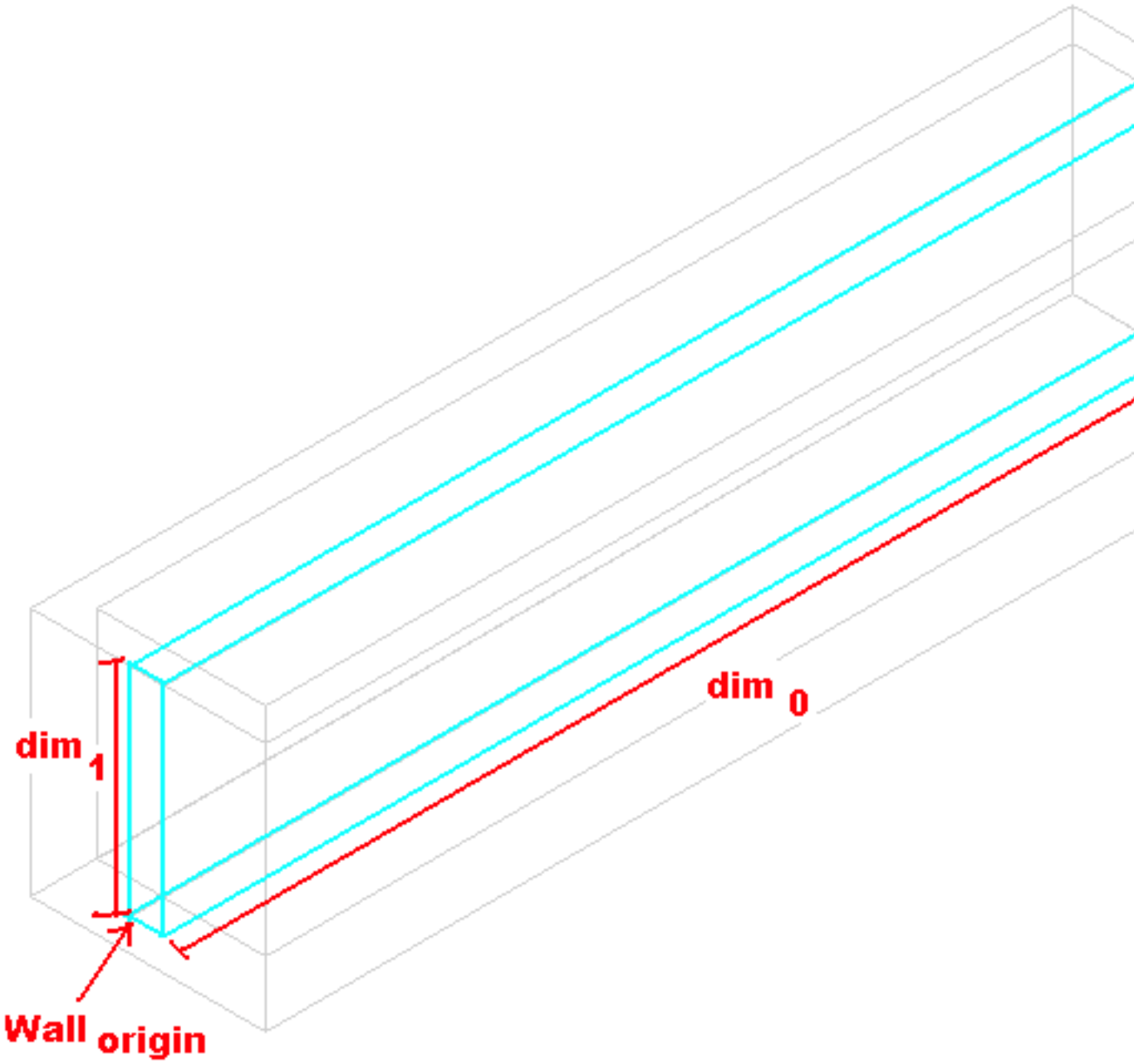
- x: $tank_{origin0}$
- y: $tank_{origin1}$ + num_{tanks}($tank_{dim1}$ + $tank_{thick}$)
- z: $tank_{origin2}$

$totalfloc_{dim}$ =

- x: $floc_{dim0}$

- y: ($numfloc_{tanks}$*$floc_{dim1}$) + (($numfloc_{tanks}$ - 1)*$tank_{thick}$)
- z: $floc_{dim2}$

$tank_{thick}$ - specifies the thickness of the tank.



Northeast Isometric View

box1 - Creates a box based on two points.

box1 <- box($wall_{origin}$,$wall_{origin}$ + $wall_{dim}$)

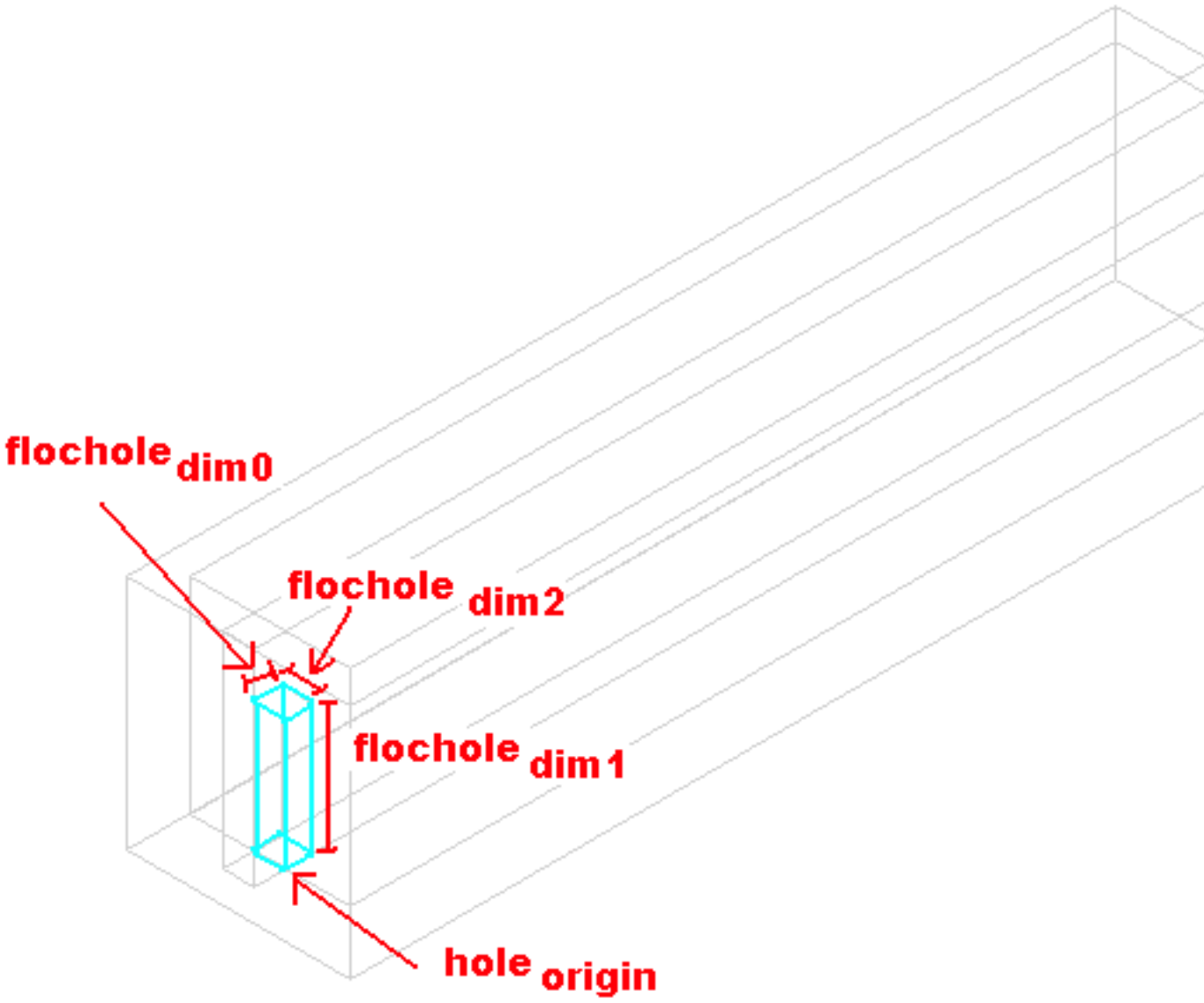
$wall_{origin}$ =

- x: $tank_{origin0}$ - $floc_{dim0}$ - $tank_{thick}$
- y: $tank_{origin1}$ + num_{tanks}($tank_{dim1}$ + $tank_{thick}$) + $floc_{dim1}$)

- z: $tank_{origin2}$

$wall_{dim}$ =

- x: $totalfloc_{dim0}$ + ($2tank_{thick}$)
- y: $tank_{thick}$
- z: $totalfloc_{dim2}$



Northeast Isometric View

box2 - Creates a box based on two points.

box2 <- box($hole_{origin}$,$hole_{origin}$ + $flochole_{dim}$)

$hole_{origin}$ =

- x: $tank_{origin0}$ - $flochole_{dim0}$
- y: $tank_{origin1}$ + num_{tanks}($tank_{dim1}$ + $tank_{thick}$) + $floc_{dim1}$
- z: $tank_{origin2}$

$flochole_{dim}$ =

- x: $tank_{dim0}$
- y: $tank_{dim1}/numfloc_{tanks}$
- z: $tank_{dim2}$

Top View

subtract0 - SubtractD subtracts one object from the other based on two points.

subtract0 <- subtractD(p1,$hole_{origin}$)

Note: p1 is a dummy variable used only in the program help section to designate the matrix below.

p1 =

- x: $hole_{origin0}$ + $flochole_{dim0}$ + $tank_{thick}$
- y: $hole_{origin1}$
- z: $hole_{origin2}$

$hole_{origin}$ =

- x: $tank_{origin0}$ - $flochole_{dim0}$
- y: $tank_{origin1}$ + num_{tanks}($tank_{dim1}$ + $tank_{thick}$) + $floc_{dim1}$
- z: $tank_{origin2}$

union1 - $Union_C$ selects objects based on two points and unites them into a single object.

union1 <- unionC(p1,p2)

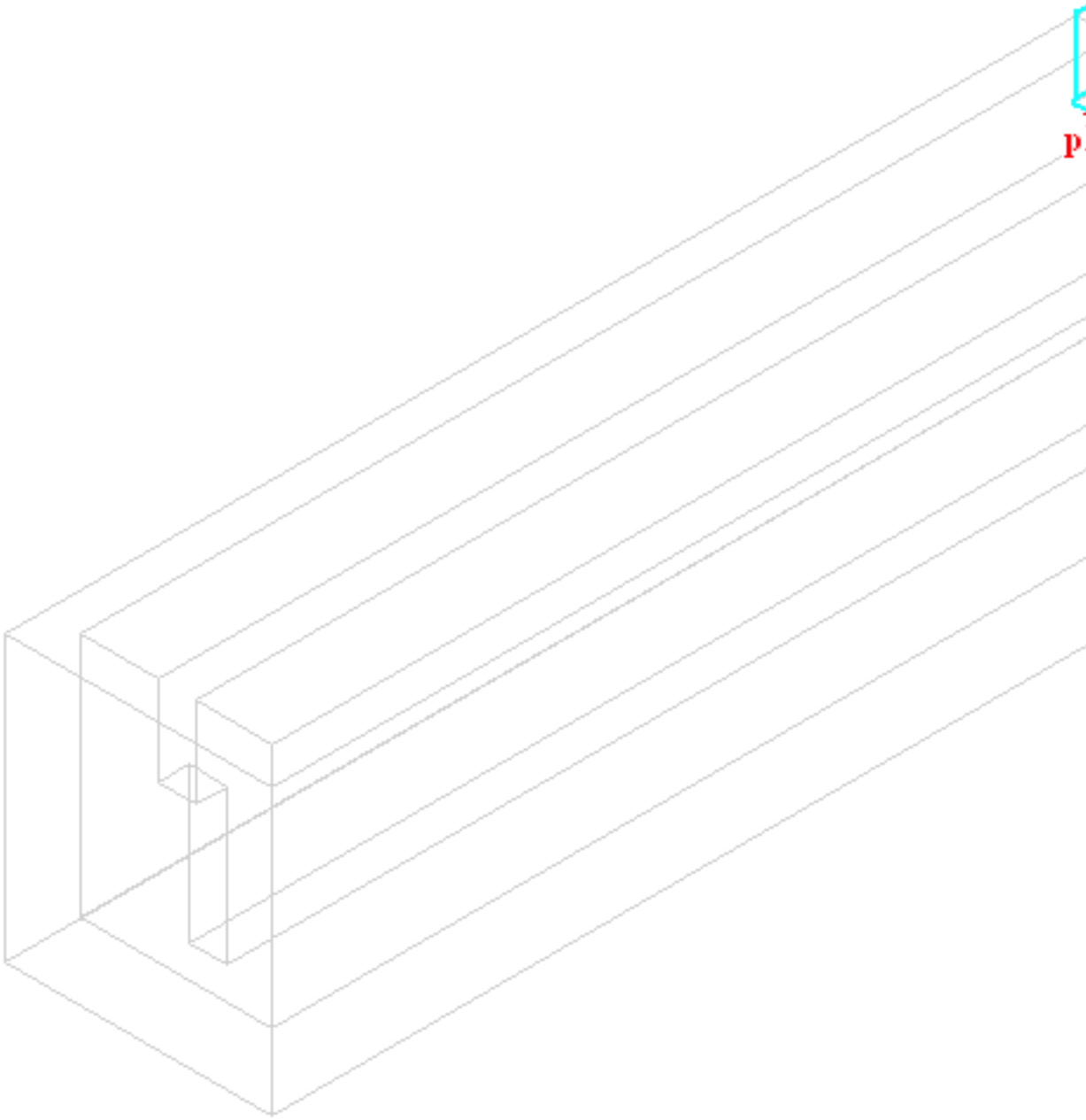
Note: p1 and p2 are dummy variables used only in the program help section to designate the matrix below.

p1 =

- x: $floc_{origin0}$ - $totalfloc_{dim0}$ - $tank_{thick}$ - zc
- y: $floc_{origin1}$ - $tank_{thick}$ - zc
- z: $floc_{origin2}$

p2 =

- x: $floc_{origin0}$ + $tank_{thick}$ + zc
- y: $floc_{origin1}$ + $totalfloc_{dim1}$ + $tank_{thick}$ + zc
- z: $floc_{origin2}$



Northeast Isometric View

box3 - Creates a box based on two points.

box3 <- box(p1,p2)

Note: p1 and p2 are dummy variables used only in the program help section to designate the matrix below.

p1 =

- x: $floc_{origin0}$ - $totalfloc_{dim0}$
- y: $floc_{origin1}$ - $tank_{thick}$
- z: $floc_{origin2}$ + $totalfloc_{dim2}$ - $channel_{dim2}$

p2 =

- x: $floc_{origin0}$ - $totalfloc_{dim0}$ + $channel_{dim1}$
- y: $floc_{origin1}$
- z: $floc_{origin2}$ + $totalfloc_{dim2}$

Top View

subtract1 - SubtractD subtracts one object from the other based on two points.

subtract1 <- subtractD($floc_{origin}$,p1)

$floc_{origin}$ =

- x: $tank_{origin0}$
- y: $tank_{origin1}$ + num_{tanks}($tank_{dim1}$ + $tank_{thick}$)
- z: $tank_{origin2}$

Note: p1 is a dummy variable used only in the program help section to designate the matrix below.

p1 =

- x: $floc_{origin0}$ - $floc_{dim0}$
- y: $floc_{origin1}$ - $tank_{thick}$
- z: $floc_{origin2}$ + $floc_{dim2}$ - $channel_{dim2}$

layerset - $Layer_{set}$ selects the layer "0".

layerset <- $layer_{set}$("0")

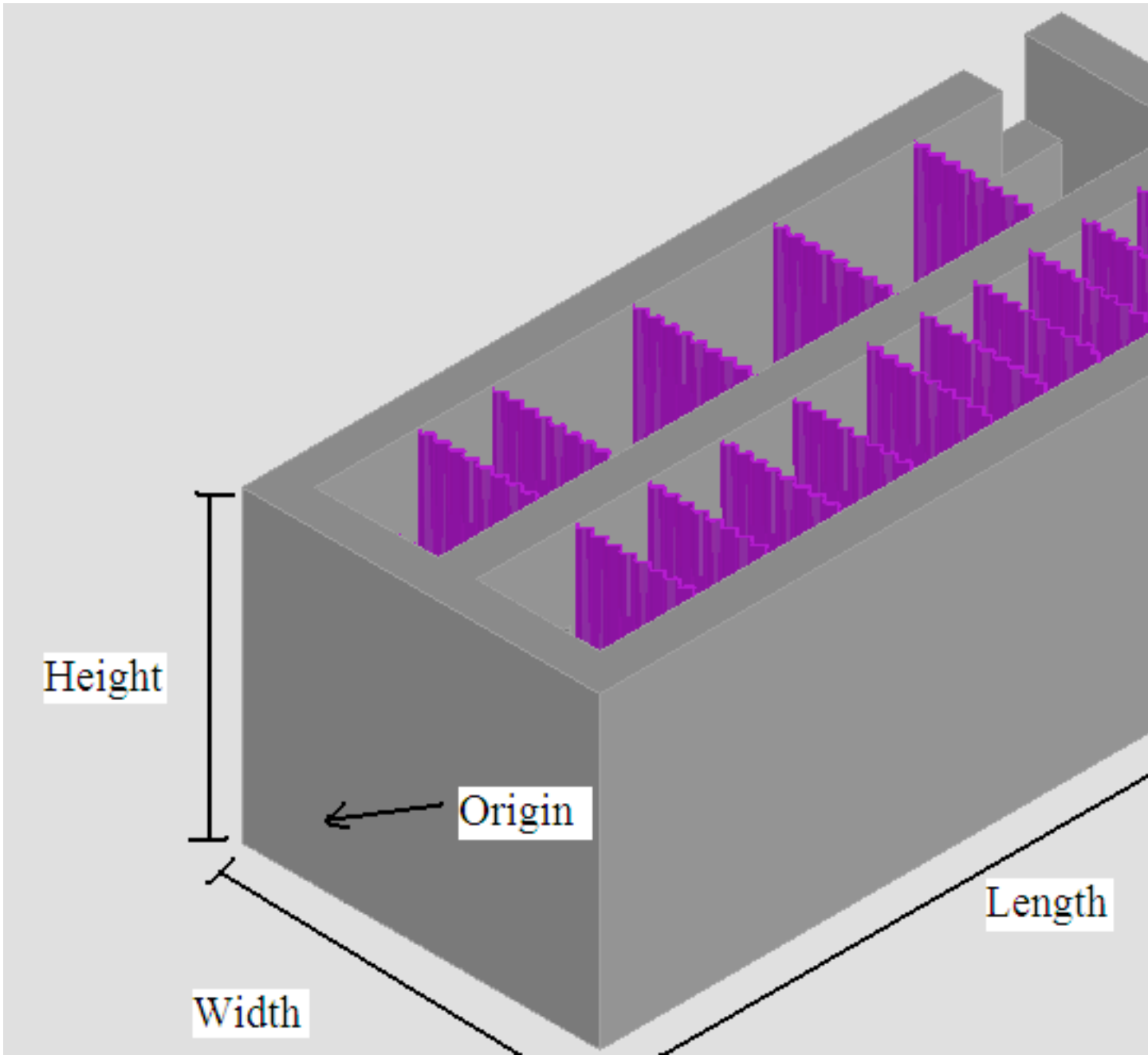
layerfreeze - $Layer_{freeze}$ locks the layer "channel" so that it cannot be edited.

layerfreeze <- layer$_{freeze}$("floctank")

AguaClara : AutoCAD Flocculation Tank Program

This page last changed on Dec 12, 2008 by ar329.

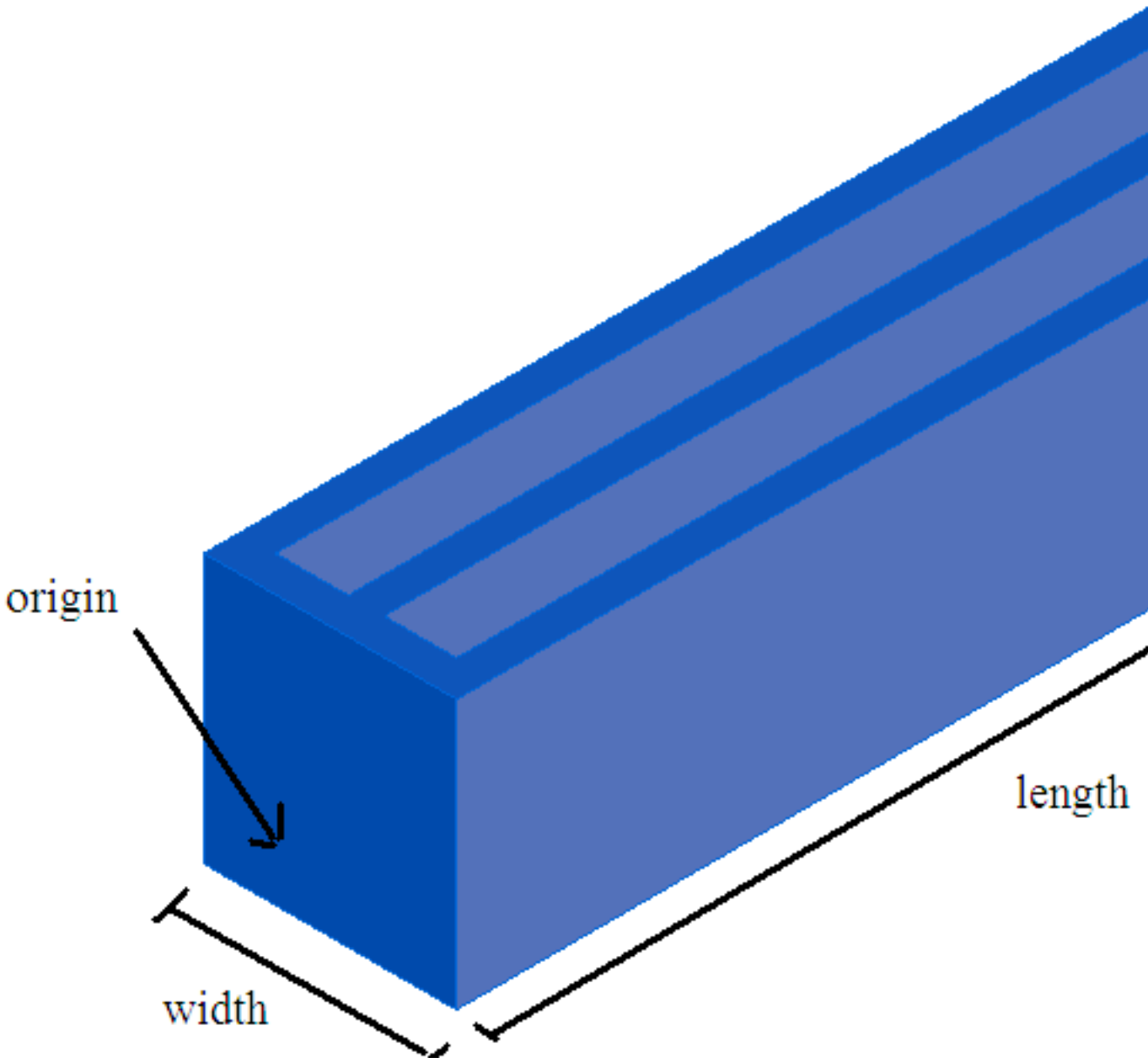
General Program Information



NorthEast Isometric View

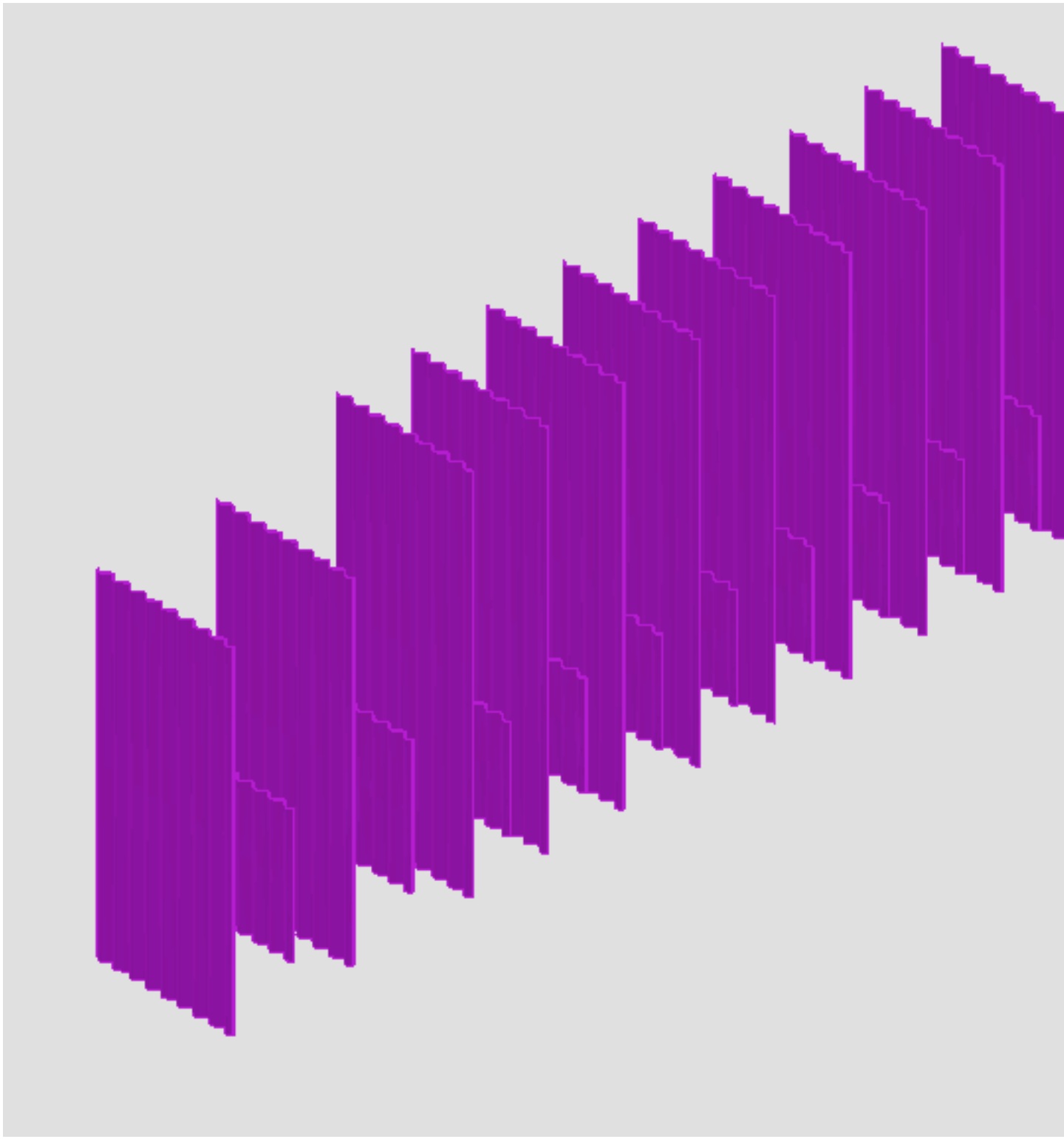
Inputs Needed to Call the Flocculation Tank Program

Inputs Defined Within the Flocculation Tank Program



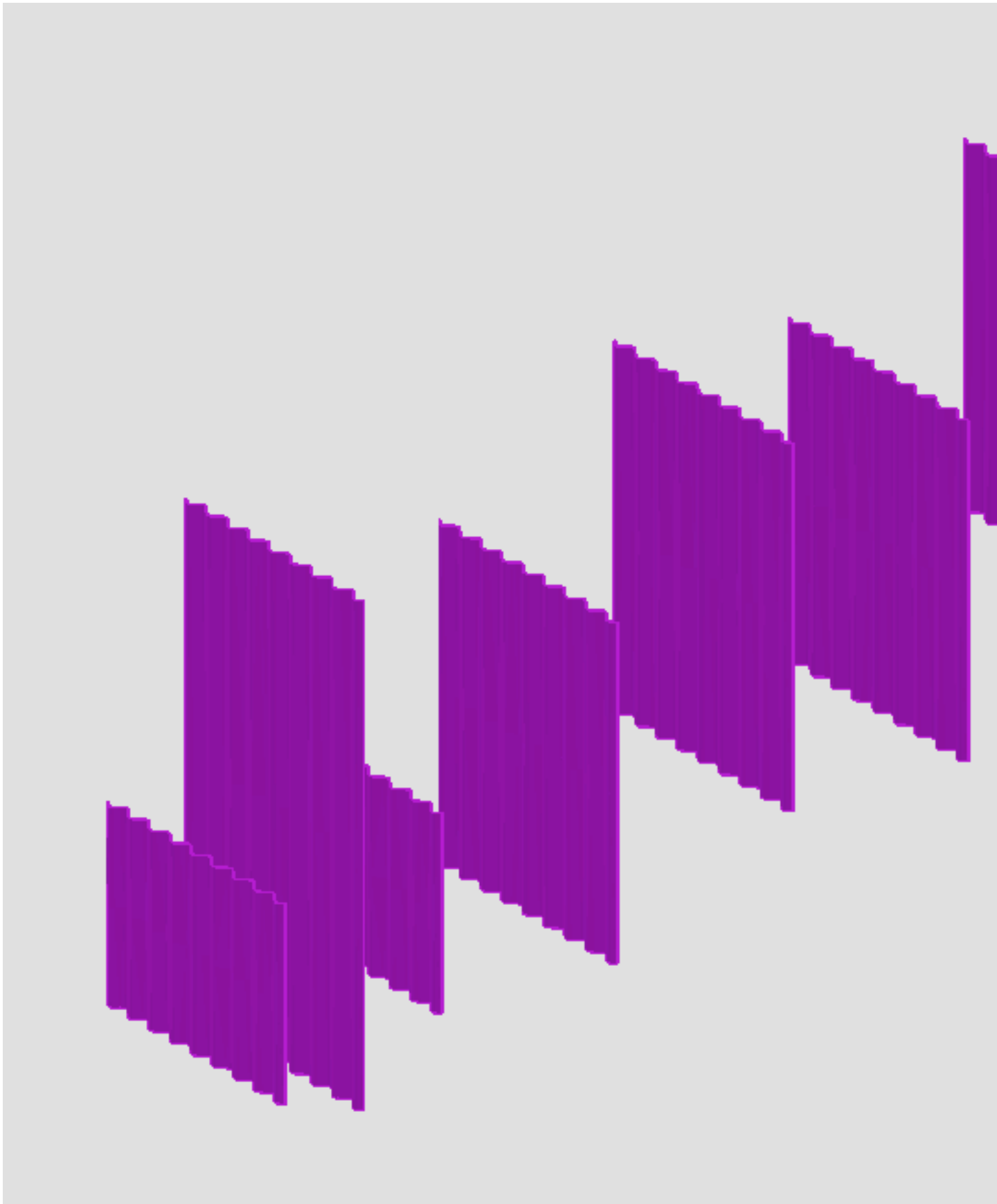
Northeast Isometric View

Floctanktankscript



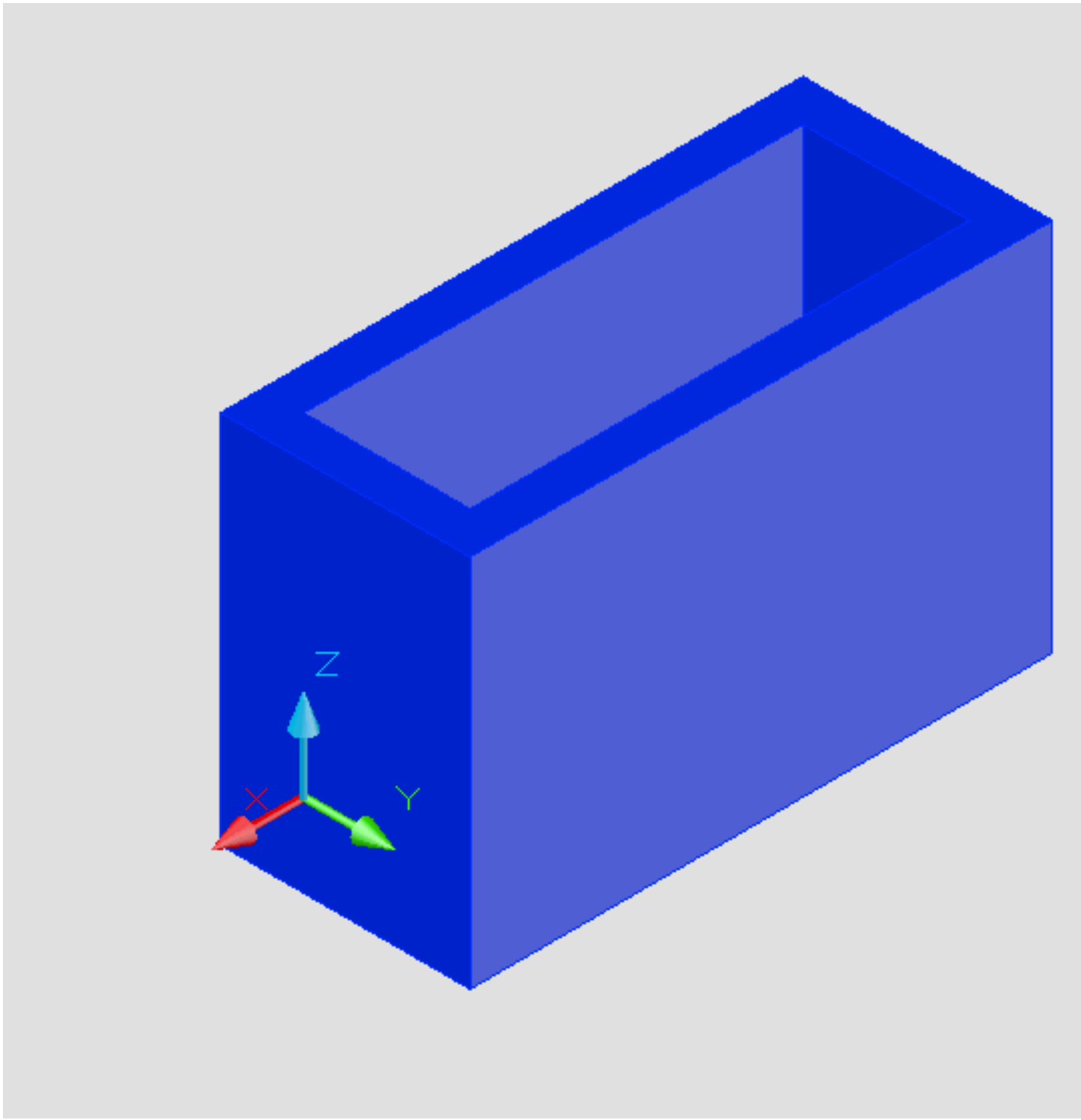
Northeast Isometric View

Floctanktankscript draws the outside walls of the flocculation tank and the median divider. It does not draw the baffles in either side of the tank.
Floctankscript2



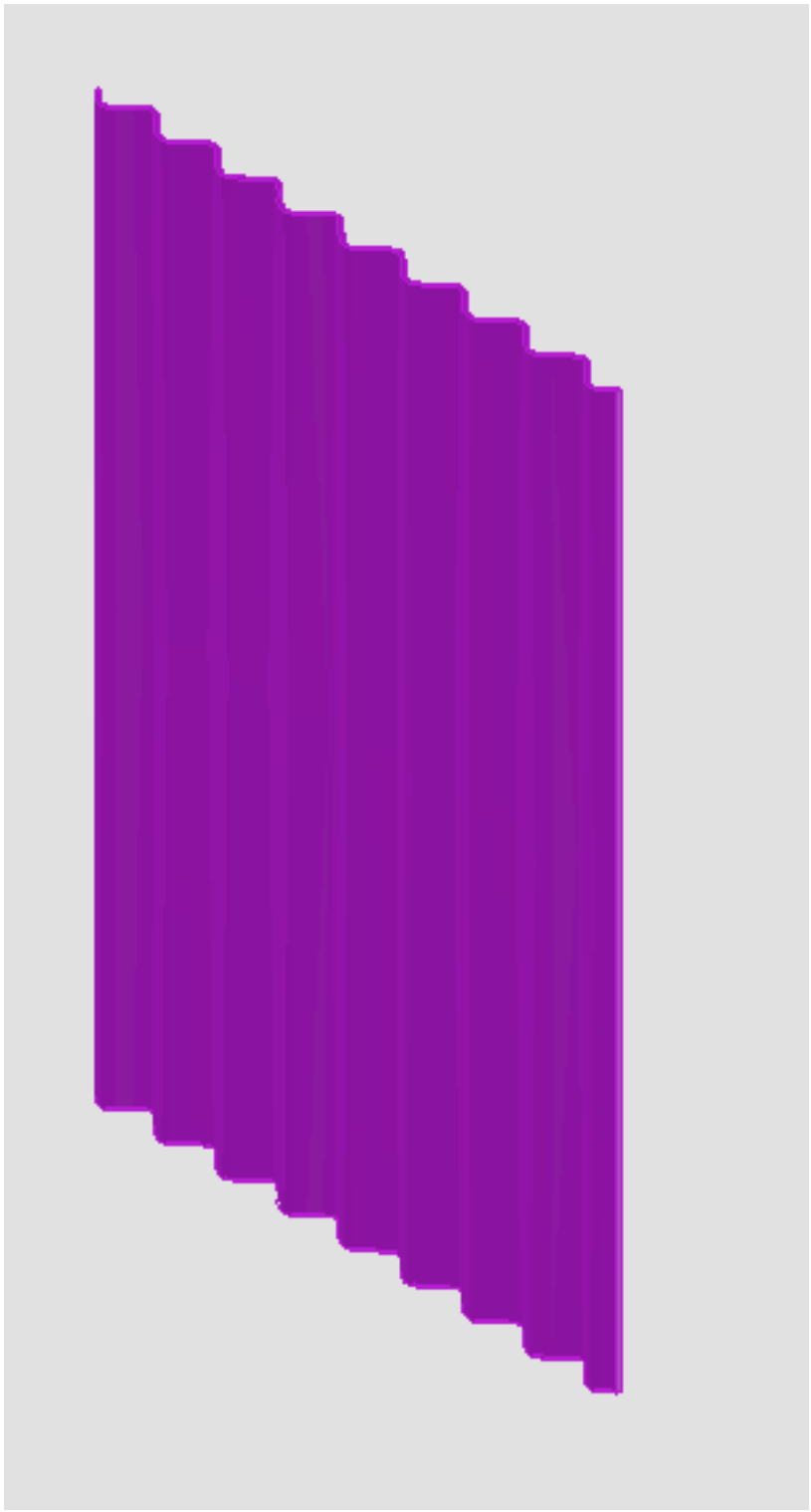
Northeast Isometric View

Floctankscript2 draws the baffles in the right side of the flocculation tank.
Floctankscript3



Northeast Isometric View

Floctankscript3 draws the baffles in the left side of the flocculation tank.
Tank
The tank program is called by the floctanktankscript to draw the outside walls of the flocculation tank.



Northeast Isometric View

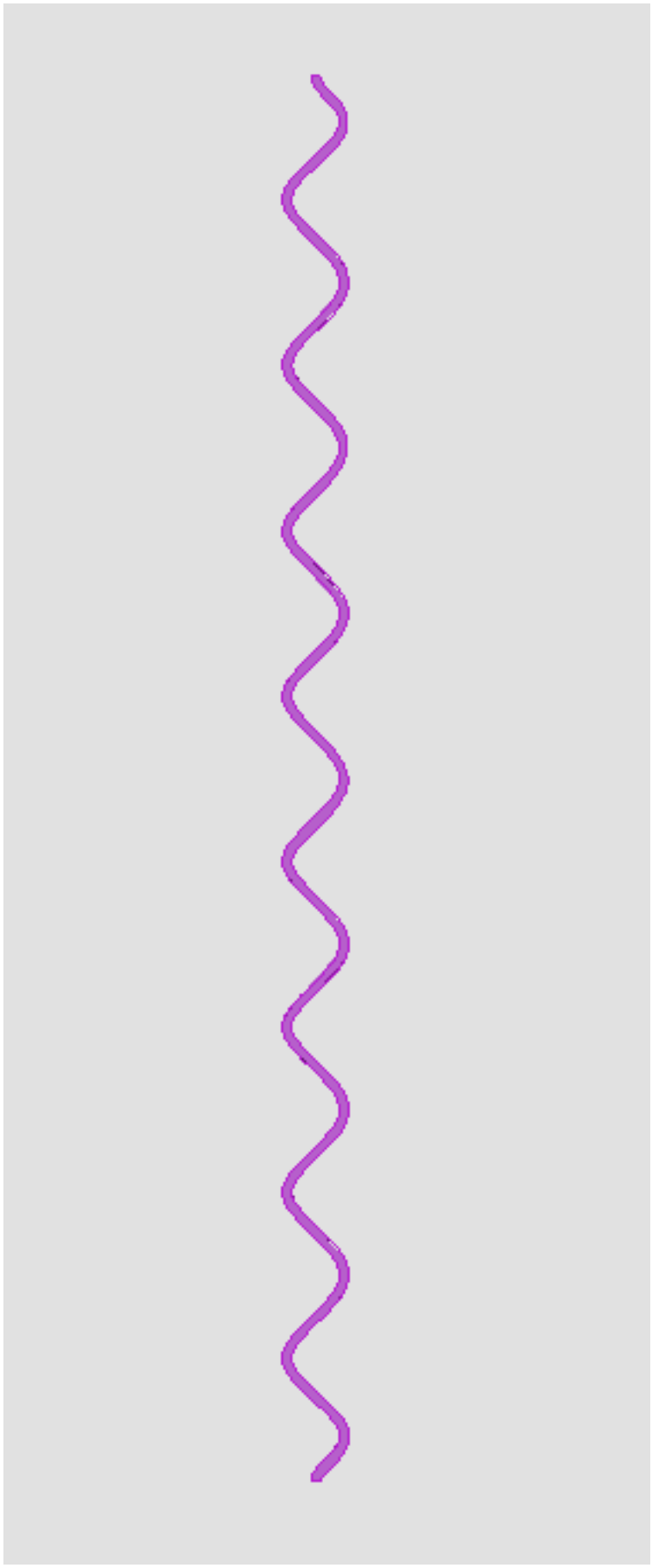
Lamina

The lamina program is called by the baffle scripts to draw lamina in the flocculation tanks.

AguaClara : AutoCAD Flocculation Tank Program - Floctankscript3

This page last changed on Nov 20, 2008 by ar329.

General Program Information



Northeast Isometric View

Technical Program

local - Calls the Lamina Program to draw a lamina at the specified origin point.

local <- lamina((baffletank2$_{origin}$ + p1),baffle$_{amp}$,baffle$_{width}$,baffletank2$_{location0,2}$,baffle$_{thick}$,baffle$_{periods}$,baffle$_{slope}$,baffle$_{num}$,baffle$_{col}$,baffle$_{x}$,baffle$_{y}$)

p1 =

- x: baffletank2{~}location0,0
- y: 0
- z: baffletank2{~}location0,1

for loop - To draw the correct number of baffles into the flocculation tank Floctankscript3 enters a loop for the range 1 to baffletank2$_{num}$ - 1. In total (baffletank2$_{num}$ - 1) baffles will be drawn. Baffle origins and lengths are drawn from the 3*8 matrix located in the inputs program.

baffle2$_{origini}$ - Determines the x,y and z coordinates of where the baffle "i" will be drawn from.

baffle2$_{origini}$ <- baffletank2$_{origin}$ + p1

baffletank2$_{origin}$ =

- x: tank$_{origin0}$ - tank$_{dim0}$
- y: tank$_{origin1}$ + num$_{tanks}$*(tank$_{dim1}$ + tank$_{thick}$)
- z: tank$_{origin2}$

p1 =

- x: baffletank2$_{locationi,0}$
- y: 0
- z: baffletank2$_{locationi,1}$

baffletank2$_{lengthi}$ - Determines the length of baffle "i" from the 3*8 matrix located in the Inputs Program.

baffletank2$_{lengthi}$ <- baffletank2$_{locationi,2}$

local - Compiles the origin and baffle lengths calculated in the for loop to draw the correct number of baffles using the Lamina Program.

local <- stack(local,lamina(baffle2$_{origini}$,baffle$_{amp}$,baffle$_{width}$,baffletank2$_{lengthi}$,baffle$_{thick}$,baffle$_{periods}$,baffle$_{slope}$,baffle$_{num}$,baffle$_{col}$,

layerset -

layerset <-

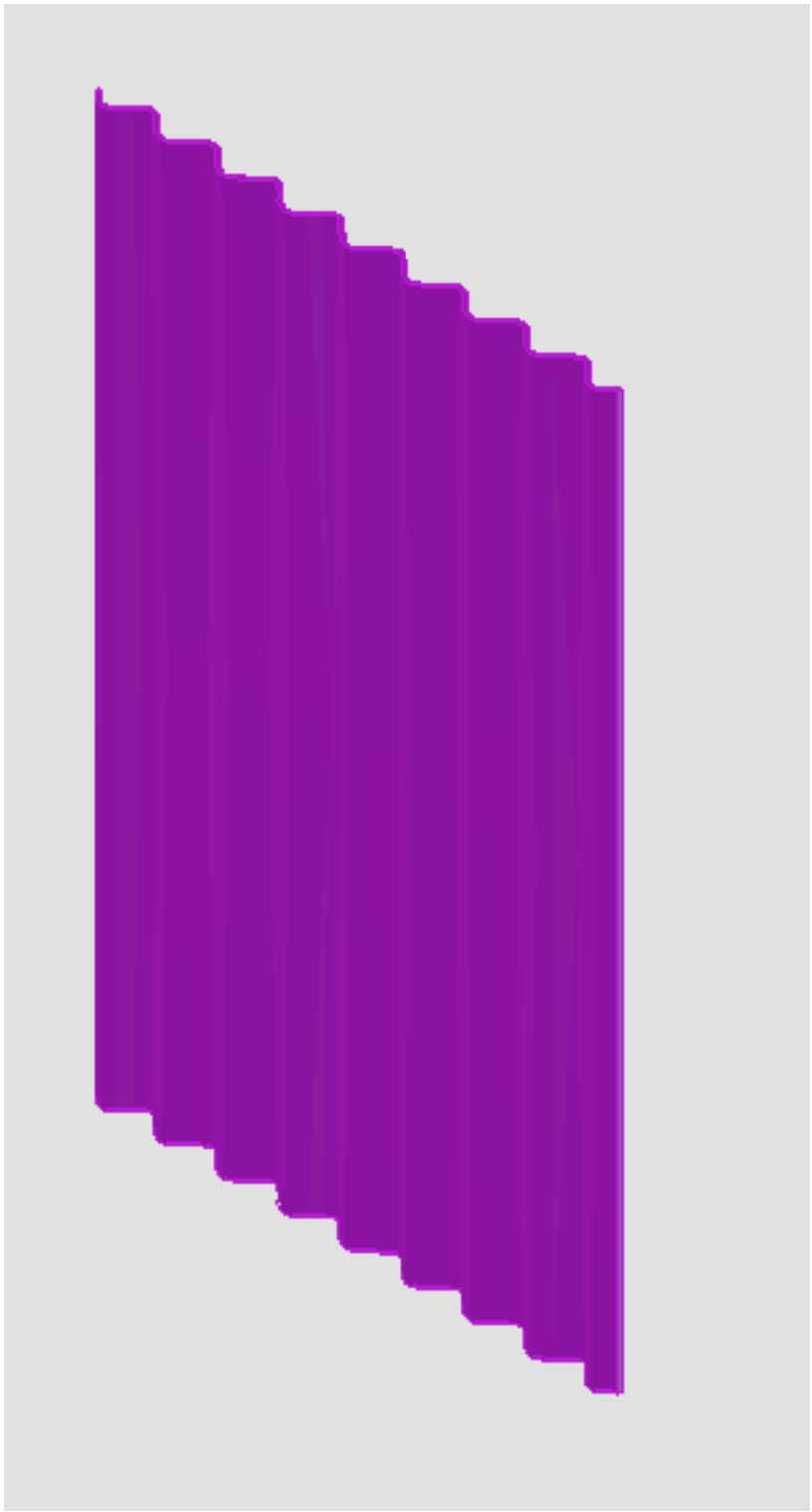
layerfreeze2 -

layerfreeze2 <-

AguaClara : AutoCAD Flocculation Tank Program - Floctankscript2

This page last changed on Nov 20, 2008 by ar329.

General Program Information



Northeast Isometric View

Technical Program Outline

local - $Layer_{new}$ creates a new light purple layer, "baffle."

local <- $layer_{new}$("baffle",ltpurple)

for loop - To draw the correct number of baffles into the flocculation tank Floctankscript2 enters a loop for the range 1 to $baffletank1_{num}$. In total $baffletank1_{num}$ baffles will be drawn. Baffle origins and lengths are drawn from the 3*22 matrix located in the Inputs Program.

$baffle1_{origin1}$ - Determines the x,y and z coordinates of where the baffle "i" will be drawn from.

$baffle1_{origin}$ <- $baffletank1_{origin}$ + p1

$baffletank1_{origin}$=

- x: $tank_{origin0}$ - $tank_{dim0}$
- y: $tank_{origin1}$ + num_{tanks}*($tank_{dim1}$ + $tank_{thick}$) + $floc_{dim1}$ + $wall_{dim1}$
- z: $tank_{origin2}$

p1 =

- x: $baffletank1_{locationi-1,0}$
- y: 0
- z: $baffletank1_{locationi-1,1}$

$baffletank1_{lengthi}$ - Determines the length of baffle "i" from the 3*22 matrix located in the Inputs Program.

$baffletank1_{lengthi}$ <- $_{baffletank1}$locationi-1,2~

local - Compiles the origin and baffle lengths calculated in the for loop to draw the correct number of baffles using the Lamina Program.

local <-
stack(local,lamina($baffle1_{origini}$,$baffle_{amp}$,$baffle_{width}$,$baffletank1_{lengthi}$,$baffle_{thick}$,$baffle_{periods}$,$baffle_{slope}$,$baffle_{num}$,$baffle_{col}$,

AguaClara : AutoCAD Floc Defined Inputs

This page last changed on Nov 18, 2008 by ar329.

Inputs Defined Within the Flocculation Tank Program

Dimensions

$totalfloc_{dim}$ =

- x: $floc_{dim0}$
- y: $(numfloc_{tanks}*floc_{dim1}) + ((numfloc_{tanks} - 1)*tank_{thick})$
- z: $floc_{dim2}$

$wall_{dim}$ =

- x: $totalfloc_{dim0} + (2tank_{thick})$
- y: $tank_{thick}$
- z: $totalfloc_{dim2}$

Origin Points

$floc_{origin}$ =

- x: $tank_{origin0}$
- y: $tank_{origin1} + num_{tanks}(tank_{dim1} + tank_{thick})$
- z: $tank_{origin2}$

$wall_{origin}$ =

- x: $tank_{origin0} - floc_{dim0} - tank_{thick}$
- y: $tank_{origin1} + num_{tanks}(tank_{dim1} + tank_{thick}) + floc_{dim1})$
- z: $tank_{origin2}$

$hole_{origin}$ =

- x: $tank_{origin0} - flochole_{dim0}$
- y: $tank_{origin1} + num_{tanks}(tank_{dim1} + tank_{thick}) + floc_{dim1}$
- z: $tank_{origin2}$

$baffletank1_{origin}$=

- x: $tank_{origin0} - tank_{dim0}$
- y: $tank_{origin1} + num_{tanks}*(tank_{dim1} + tank_{thick}) + floc_{dim1} + wall_{dim1}$
- z: $tank_{origin2}$

$baffletank2_{origin}$ =

- x: $tank_{origin0} - tank_{dim0}$
- y: $tank_{origin1} + num_{tanks}*(tank_{dim1} + tank_{thick})$
- z: $tank_{origin2}$

AguaClara : AutoCAD Floc Called Inputs

This page last changed on Dec 10, 2008 by ar329.

Inputs Needed to Call the Flocculation Tank Program

$numfloc_{tanks}$ - Indicates the number of flocculation tanks specified in the plant design.

$floc_{inleth}$ - Indicates the height of the floc inlet pipe.

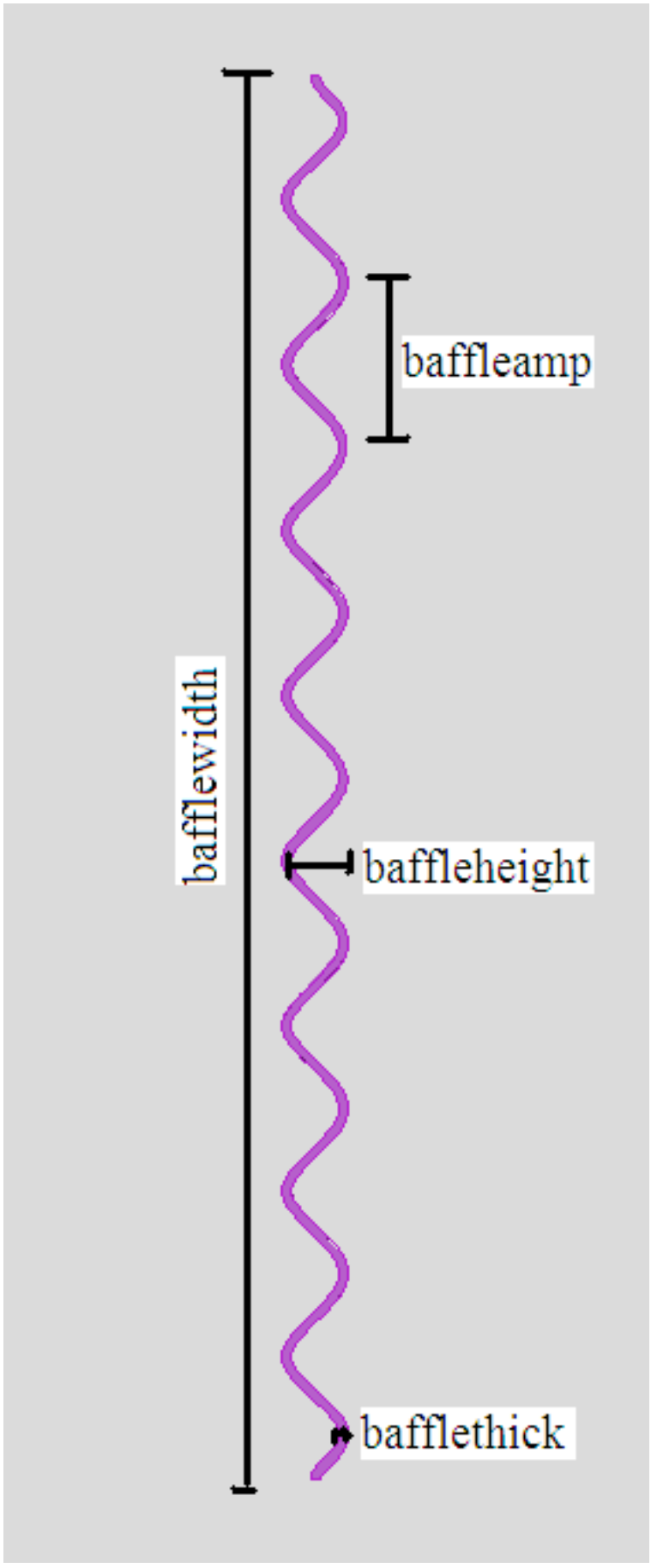
$flochole_{dim}$ - A 3*1 matrix with x,y,z user defined inputs specifying the dimensions of the flochole.

$floc_{dim}$ - A 3*1 matrix specifying the x,y,z dimensions of the floc tank.

- x: $tank_{dim0}$
- y: $tank_{dim1}/numfloc_{tanks}$
- z: $tank_{dim2}$

$baffletank1_{location}$ - A 22*3 matrix that specifies the x,y,z coordinates of the base points from which each baffle in the tank will be drawn.

$baffletank1_{num}$ - Specifies the number of baffles that will be drawn in tank 1.



Top View

$baffle_{amp}$ - Specifies the amplitude of each baffle.

$baffle_{width}$ - Specifies the width of each individual baffle.

$baffle_{periods}$ - Specifies the period of each baffle

$baffle_{num}$ -

$baffle_{x}$ -

$baffle_{thick}$ - Specifies the thickness of each baffle.

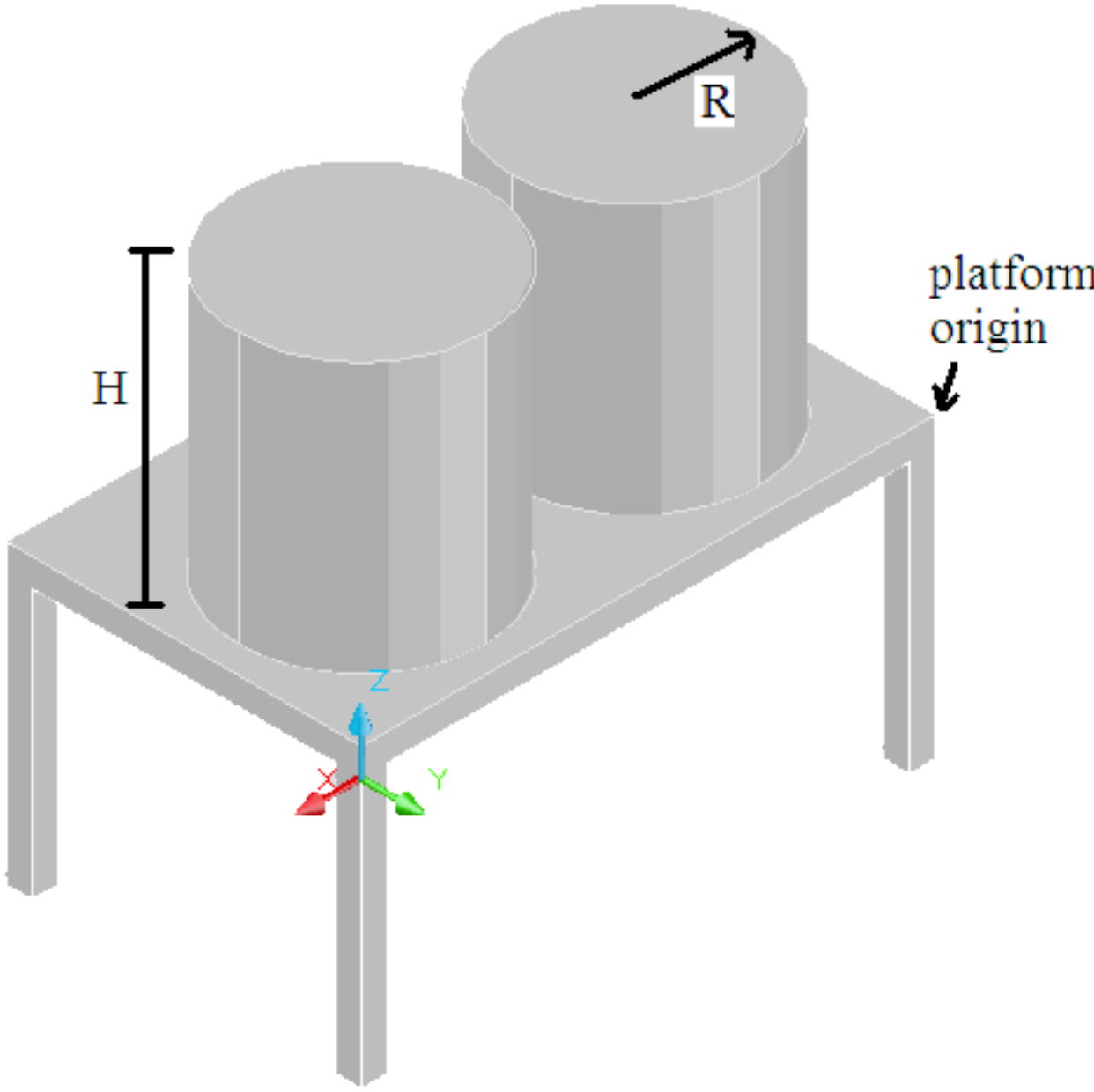
$baffle_{slope}$ - Specifies the slope of the baffles with respect to the bottom of the tank.

$baffle_{col}$ -

$baffle_{y}$ -

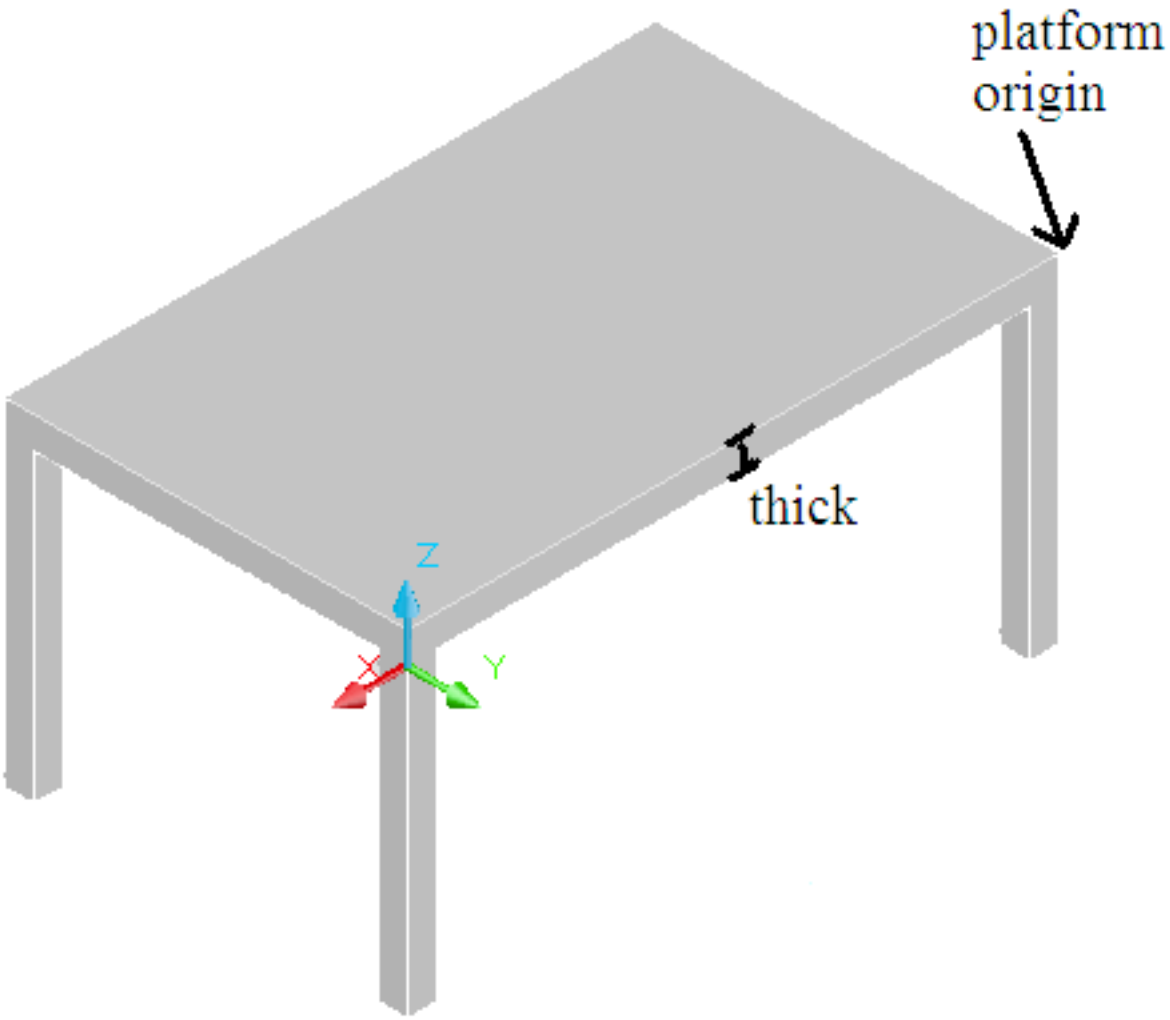
This page last changed on Dec 17, 2008 by ar329.

AutoCAD Chemical Stock Tank Program



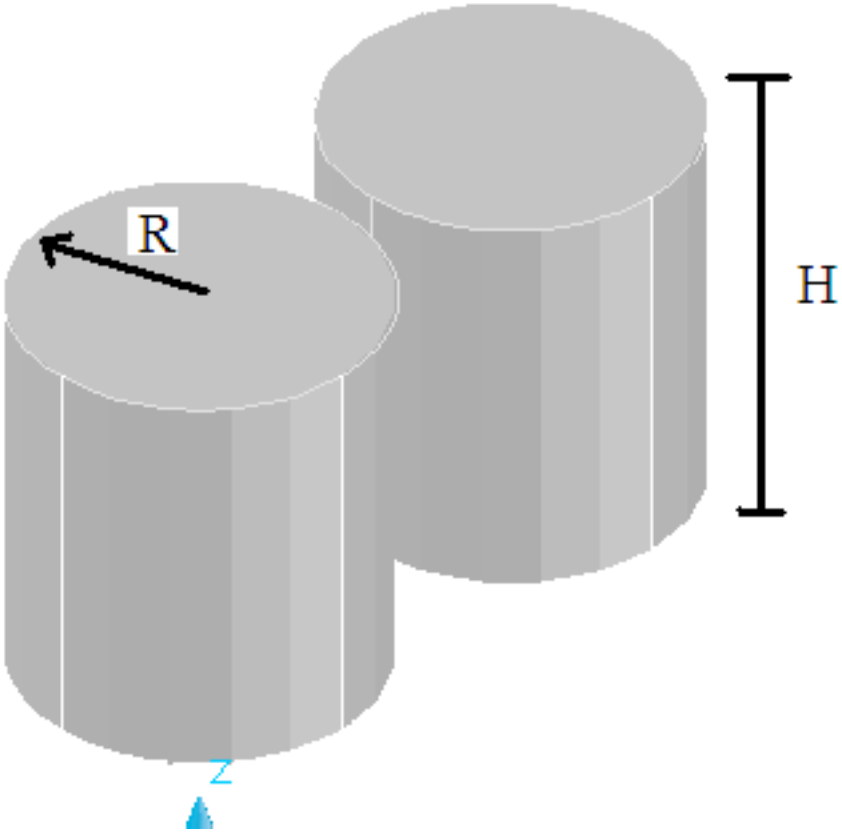
NE Isometric View

The chemical stock tank drawing program consists of two separate programs. the first draws the chemical stock tank platforms while the second draws the stock tank barrels that sit on this platform. The descriptions of the programs are given individually below



NE Isometric View

[Chemical Stock Tank Platform]

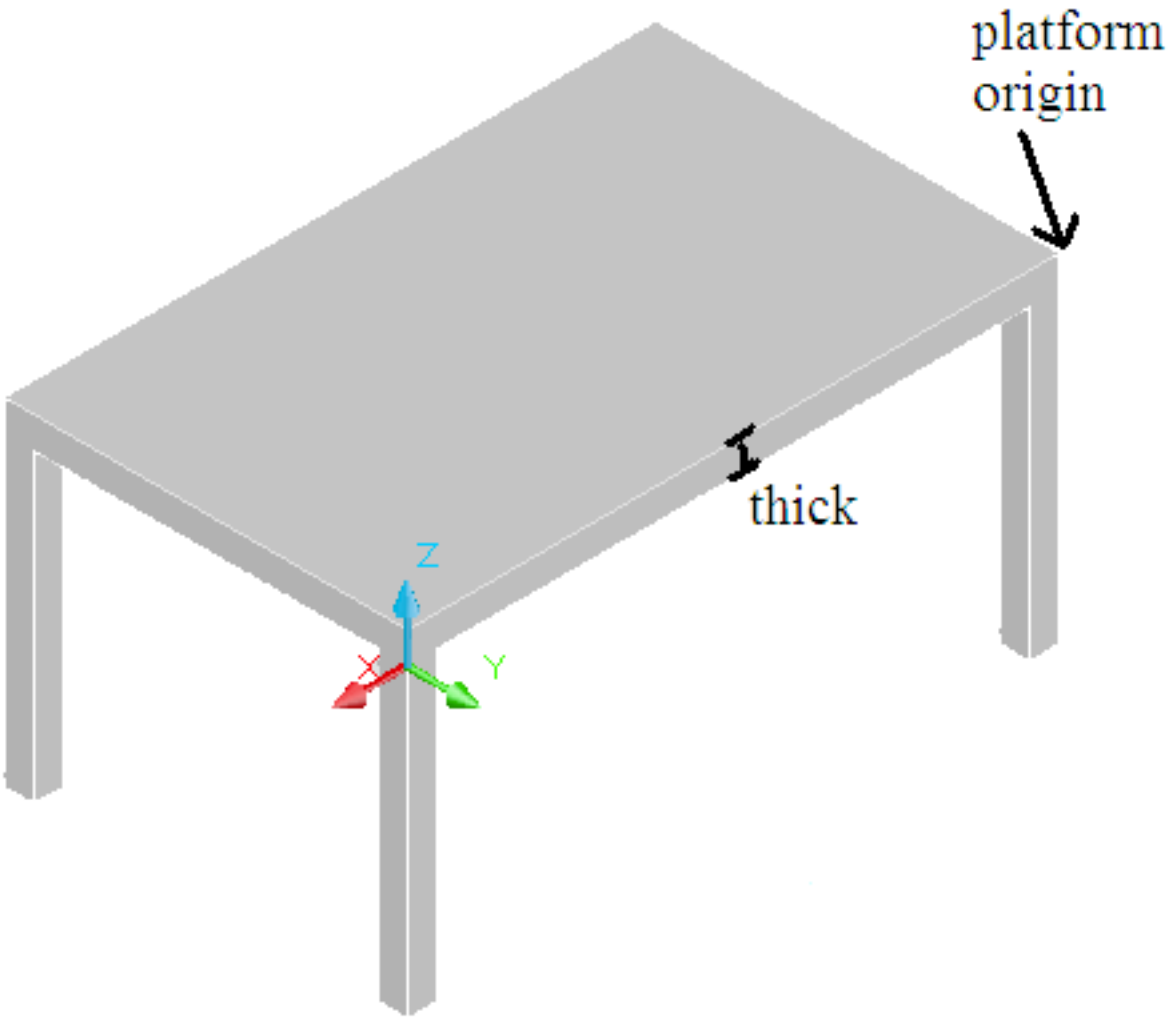


NE Isometric View

[Chemical Stock Tank Barrels]

This page last changed on Dec 11, 2008 by ar329.

General Program Information



NE Isometric View

Input Definitions

Inputs Needed to Call the Chemical Stock Tank Function

- origin - a 3*1 matrix with user defined x,y,z positions corresponding to origin. The origin is located at the top right corner of the platform.
- disp - displacement between the edge of the drum containing the chemical stock solution (sometimes also refered to as the chemical stock barrel) and the edge of the platform.
- thick - specifies the thickness of the platform.
- walkway - the width of a walkway space on the platform, so that the plant operater can walk on the platform next to the chemical stock drums/barrels
- $column_{dim}$ - a 3*1 matrix with the x,y,z dimensions of the columns.

- R - chemical stock drum radius/ chemical stock barrel radius

Inputs Defined within the Chemical Stock Tank Function

$platform_{origin}$ =

*x : $origin_0$
*y : $origin_1$
*Z : $origin_2$

$platform_{dim}$ =

*x : 2(2R) + (3*disp)
*y : (2R) + walkway + disp
*Z : thick

$plat_{origin}$ =

*x : $platform_{origin0}$
*y : $platform_{origin1}$
*z : $platform_{origin2}$

$plat_{point}$ =

*x : $platform_{origin0}$ - $platform_{dim0}$
*y : $platform_{origin1}$ - $platform_{dim1}$
*z : $platform_{origin2}$ + $platform_{dim2}$

$coll_{origin}$ =

*x : $platform_{origin0}$ - $platform_{dim0}$
*y : $platform_{origin1}$
*z : $platform_{origin2}$

$coll_{point}$ =

*x : $column1_{origin0}$ + $column_{dim0}$
*y : $column1_{origin1}$ - $column_{dim1}$
*z : $column1_{origin2}$ - $column_{dim2}$

$col2_{origin}$ =

*x : $platform_{origin0}$
*y : $platform_{origin1}$
*z : $platform_{origin2}$

$col2_{point}$ =

*x : $column2_{origin0}$ - $column_{dim0}$
*y : $column2_{origin1}$ - $column_{dim1}$
*z : $column2_{origin2}$ - $column_{dim2}$

$col3_{origin}$ =

*x : $platform_{origin0}$ - $platform_{dim0}$
*y : $platform_{origin1}$ - $platform_{dim1}$
*z : $platform_{origin2}$

$col3_{point}$ =

*x : $column3_{origin0}$ + $column_{dim0}$
*y : $column3_{origin1}$ + $column_{dim1}$

*z : $column3_{origin2}$ - $column_{dim2}$

$col4_{origin}$ =

*x : $platform_{origin0}$
*y : $platform_{origin1}$ - $platform_{dim1}$
*z : $platform_{origin2}$

$col4_{point}$ =

*x : $column4_{origin0}$ - $column_{dim0}$
*y : $column4_{origin1}$ + $column_{dim1}$
*z : $column4_{origin2}$ - $column_{dim2}$

Technical Program Outline

Top View

box1 - creates a box that will serve as the platform surface.

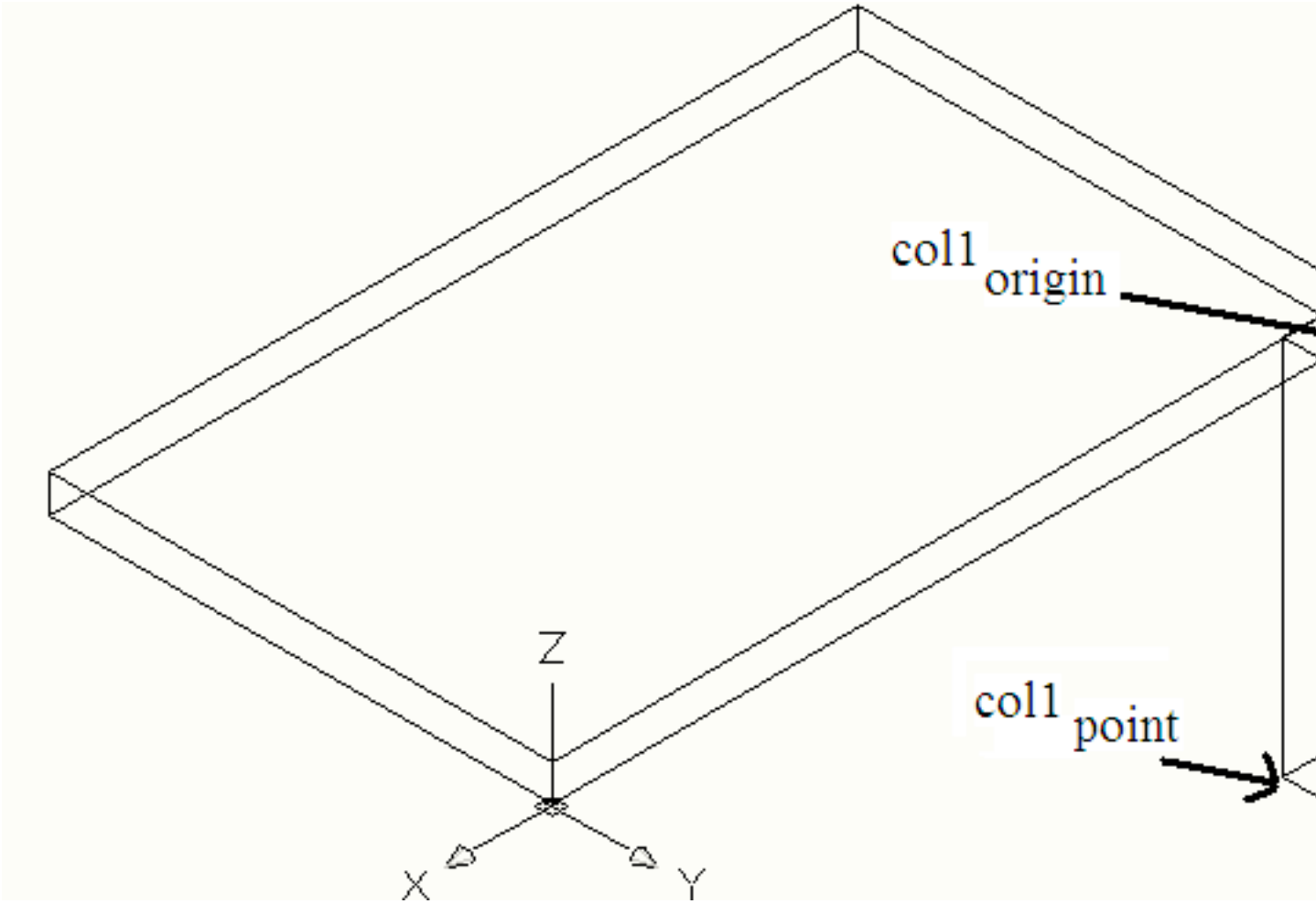
box1 <-- box($plat_{origin}$,$plat_{point}$)

$plat_{origin}$ =

*x : $platform_{origin0}$
*y : $platform_{origin1}$
*z : $platform_{origin2}$

$plat_{point}$ =

*x : $platform_{origin0}$ - $platform_{dim0}$
*y : $platform_{origin1}$ - $platform_{dim1}$
*z : $platform_{origin2}$ + $platform_{dim2}$



NorthEast Isometric View

box2 - creates a box that will serve as the upper left column holding up the platform when the structure is viewed from topview.

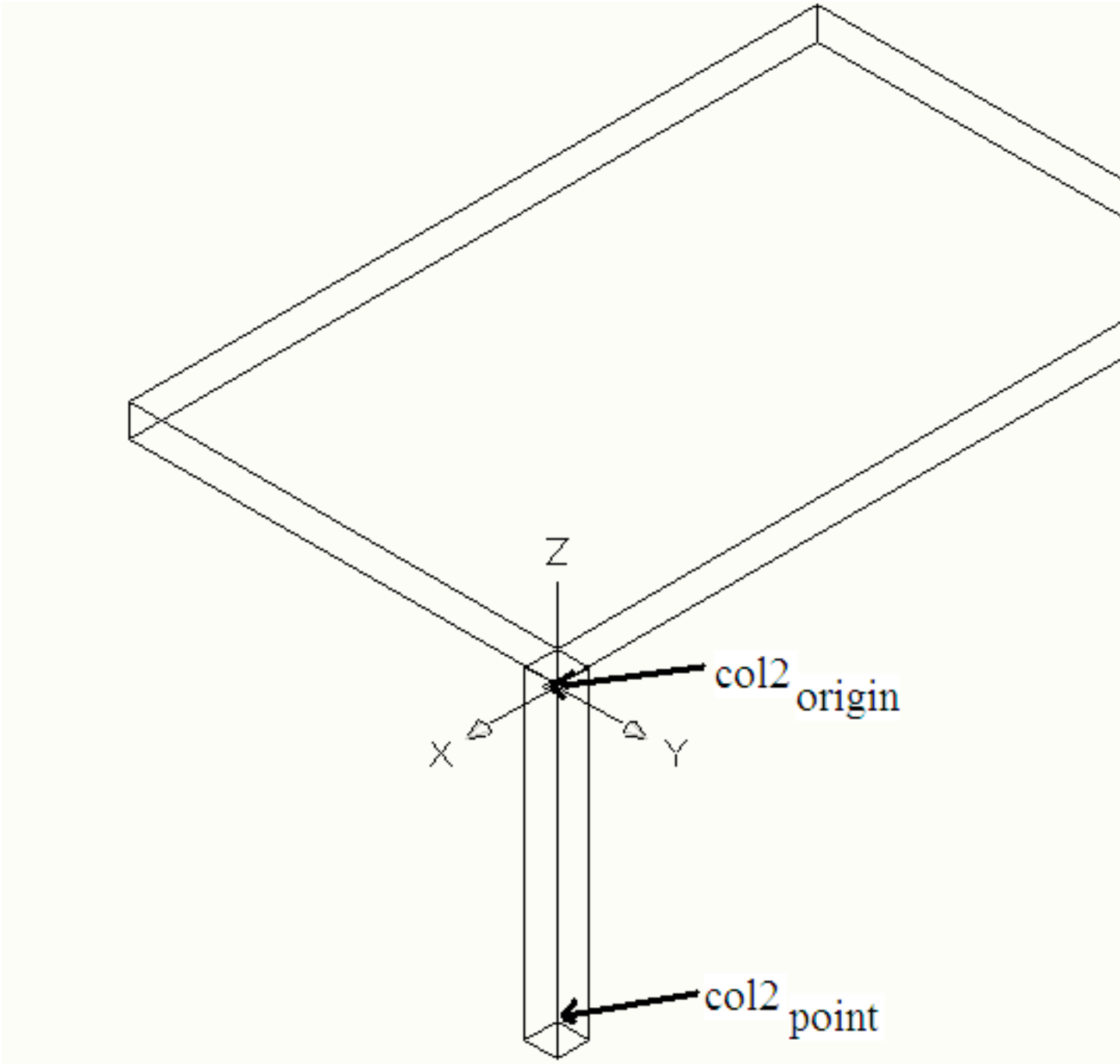
box2 <-- box($col1_{origin}$,$col1_{point}$)

$coll_{origin}$ =

*x : $platform_{origin0}$ - $platform_{dim0}$
*y : $platform_{origin1}$
*z : $platform_{origin2}$

$coll_{point}$ =

*x : $column1_{origin0}$ + $column_{dim0}$
*y : $column1_{origin1}$ - $column_{dim1}$
*z : $column1_{origin2}$ - $column_{dim2}$



NorthEast Isometric View

box3 - creates a box that will serve as the upper right column holding up the platform when the structure is viewed from topview.

box3 <-- box($col2_{origin}$,$col2_{point}$)

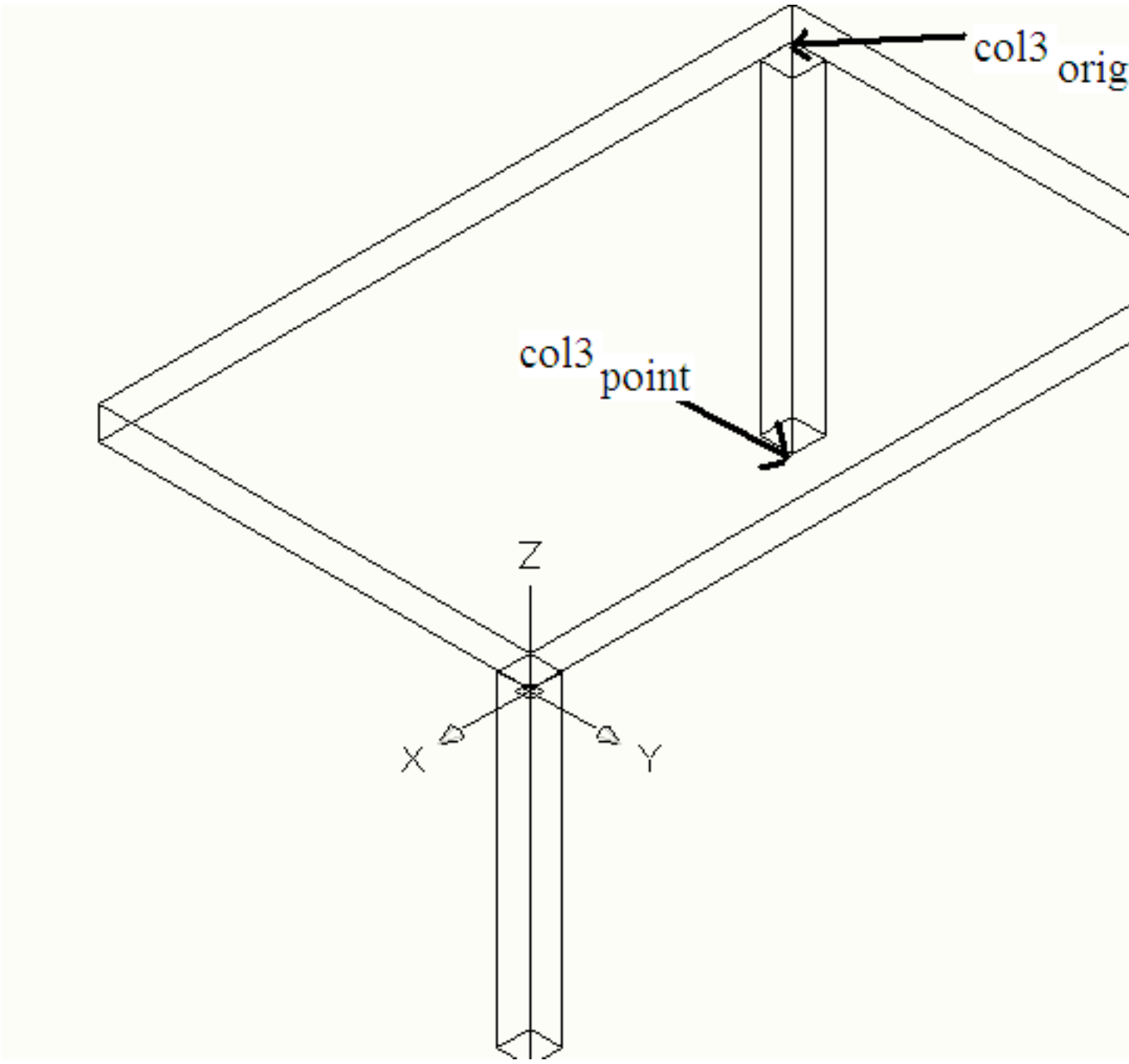
$col2_{origin}$ =

*x : $platform_{origin0}$
*y : $platform_{origin1}$
*z : $platform_{origin2}$

$col2_{point}$ =

*x : $column2_{origin0}$ - $column_{dim0}$
*y : $column2_{origin1}$ - $column_{dim1}$

*z : $column2_{origin2}$ - $column_{dim2}$



NorthEast Isometric View

box4 - creates a box that will serve as the lower left column holding up the platform when the structure is viewed from topview.

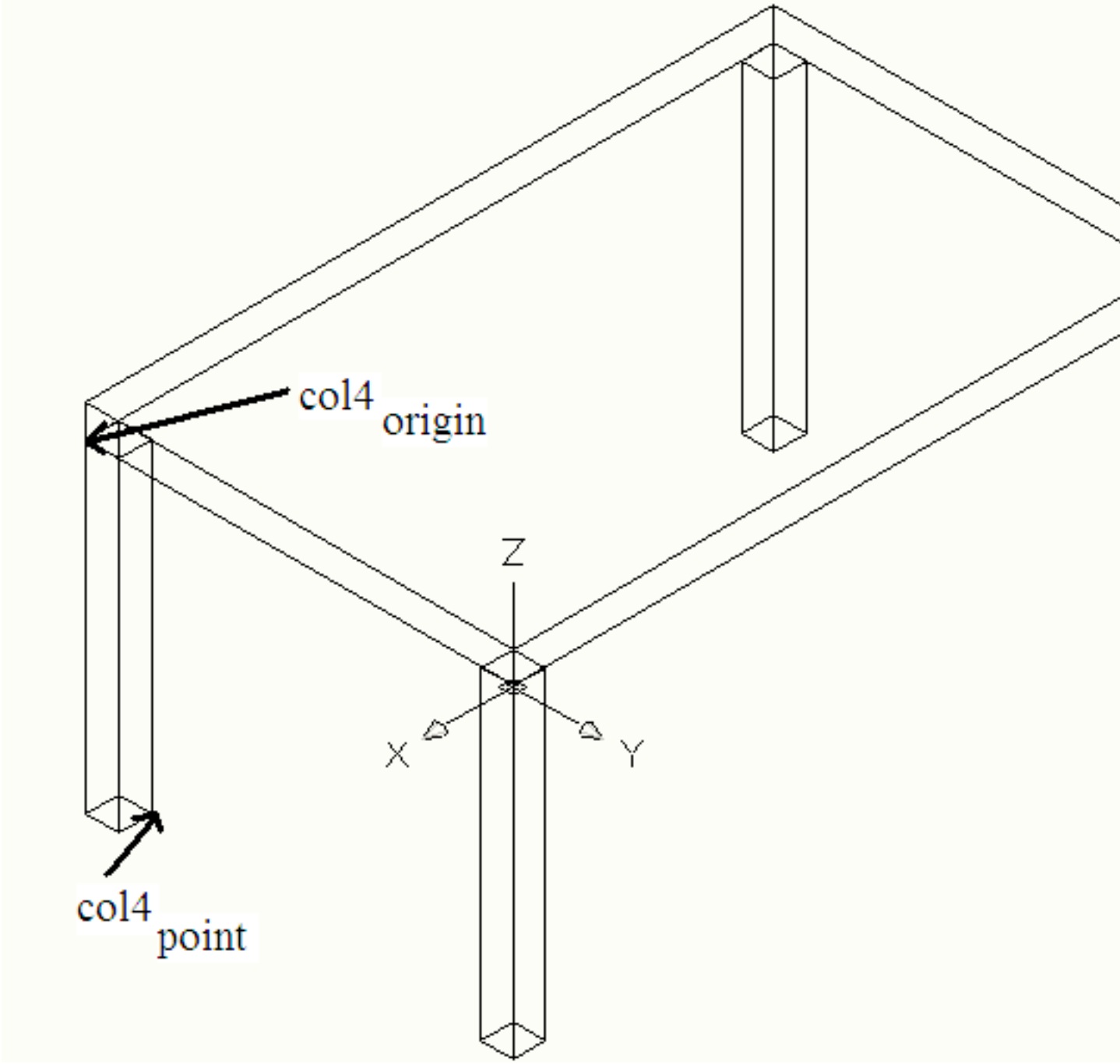
box4 <-- box($col3_{origin}$,$col3_{point}$)

$col3_{origin}$ =

*x : $platform_{origin0}$ - $platform_{dim0}$
*y : $platform_{origin1}$ - $platform_{dim1}$
*z : $platform_{origin2}$

$col3_{point}$ =

*x : $column3_{origin0}$ + $column_{dim0}$
*y : $column3_{origin1}$ + $column_{dim1}$
*z : $column3_{origin2}$ - $column_{dim2}$



NorthEast Isometric View

box5 - creates a box that will serve as the lower left column holding up the platform when the structure is viewed from topview.

box5 <-- box($col4_{origin}$,$col4_{point}$)

$col4_{origin}$ =

*x : $platform_{origin0}$
*y : $platform_{origin1}$ - $platform_{dim1}$
*z : $platform_{origin2}$

$col4_{point}$ =

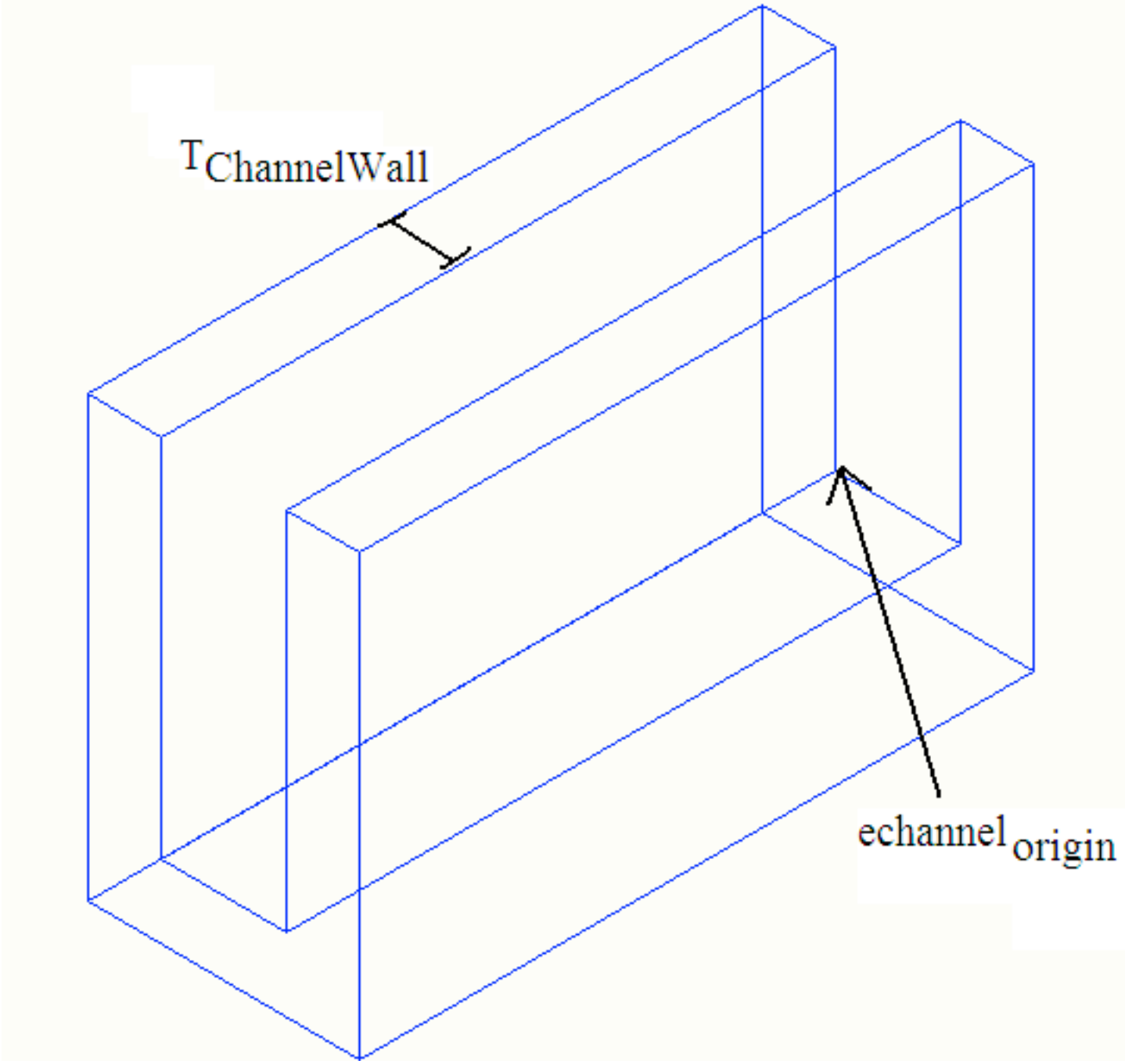
*x : $column4_{origin0}$ - $column_{dim0}$
*y : $column4_{origin1}$ + $column_{dim1}$
*z : $column4_{origin2}$ - $column_{dim2}$
union1 - $union_{allA}$is used to union the four columns and the top of the platform to create a single solid platform structure upon which the chemical stock tanks can be placed.

This page last changed on Dec 17, 2008 by ar329.

Exit Channel Drawing Script

layer2 - $Layer_{new}$ creates a new light grey layer, "echannel."

layer2 <- $layer_{new}$("echannel",ltgrey)



Northeast Isometric View

channel1 - Calls the Channel Program to draw to tank channel based on given inputs.

channel1 <- ChannelDrawing($echannel_{origin}$,p1,$T_{ChannelWall}$)

Note: $echannel_{origin}$ is calculated through a for loop.

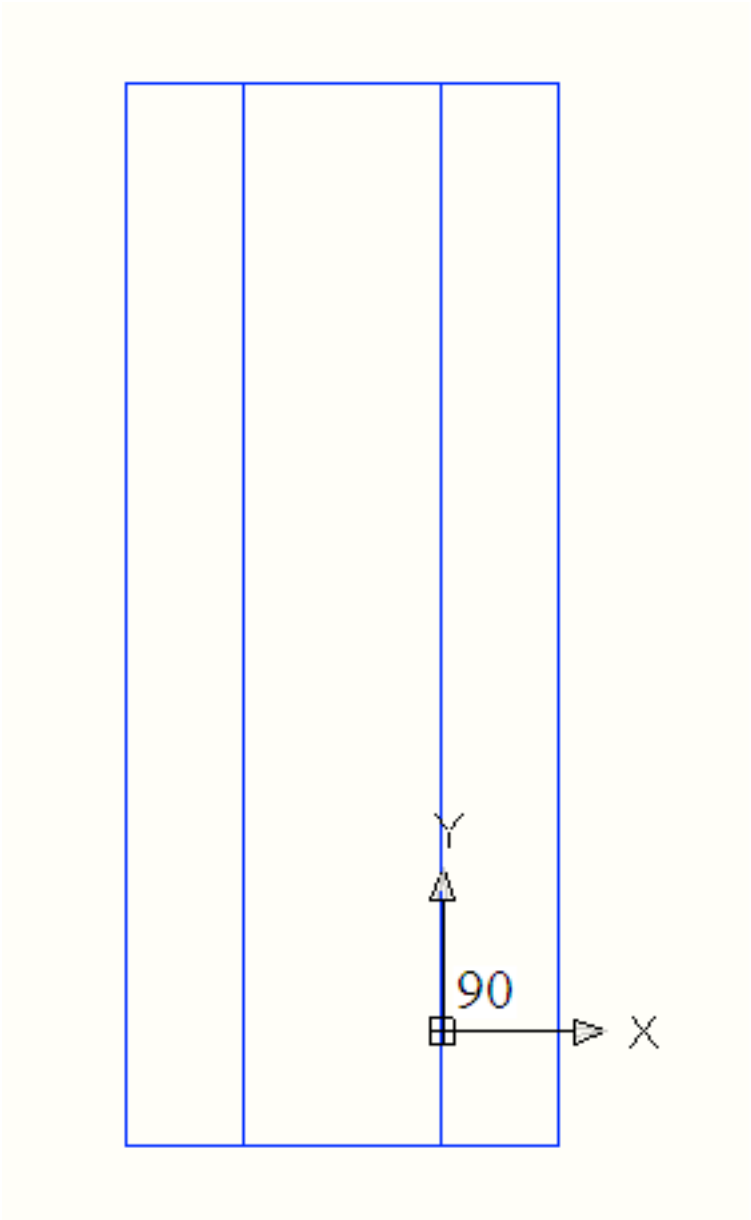
$echannel_{origin}$ =

- if layout1:
  - x: 0
  - y: 0
  - z: 0
- if layout2:
  - x: $tank_{origin0}$
  - y: $tank_{origin1}$ - $T_{PlantWall}$
  - z: $tank_{origin2}$ + H_{Sed} - $H_{EChannel}$
- if layout3:
  - x: $tank_{origin0}$ - L_{Sed} + $W_{Channel}$ + $T_{ChannelWall}$ + $W_{EChannel}$
  - y: $tank_{origin1}$ - $T_{PlantWall}$
  - z: $tank_{origin2}$ + H_{Sed} - $H_{EChannel}$
- if layout4:
  - x: $tank_{origin0}$ - L_{Sed} + $W_{ChannelInlet}$ + $T_{ChannelWall}$ + $W_{EChannel}$
  - y: $tank_{origin1}$ - $T_{PlantWall}$
  - z: $tank_{origin2}$ + H_{Sed} - $H_{EChannel}$

p1 =

- x: W_{Sed} + $2T_{PlantWall}$
- y: $W_{ExitChannel}$
- z: $H_{ExitChannel}$

$T_{ChannelWall}$ = 0.15m



Top View

rotate1 - Rotate turns the object based on a given point and rotation angle.

rotate1 <- rotate($echannel_{origin}$,90)

Note: $echannel_{origin}$ is calculated through a for loop.

$echannel_{origin}$ =

- if layout1:
  - x: 0
  - y: 0
  - z: 0
- if layout2:
  - x: $tank_{origin0}$

    - y: $tank_{origin1}$ - $T_{PlantWall}$
    - z: $tank_{origin2}$ + H_{Sed} - $H_{EChannel}$

- if layout3:
    - x: $tank_{origin0}$ - L_{Sed} + $W_{Channel}$ + $T_{ChannelWall}$ + $W_{EChannel}$
    - y: $tank_{origin1}$ - $T_{PlantWall}$
    - z: $tank_{origin2}$ + H_{Sed} - $H_{EChannel}$

- if layout4:
    - x: $tank_{origin0}$ - L_{Sed} + $W_{ChannelInlet}$ + $T_{ChannelWall}$ + $W_{EChannel}$
    - y: $tank_{origin1}$ - $T_{PlantWall}$
    - z: $tank_{origin2}$ + H_{Sed} - $H_{EChannel}$

      bigunion - $Union_{allA}$ selects all the objects currently in the workspace and joins them into one single object.

bigunion <- $union_{allA}$

layerset - $Layer_{set}$ selects the layer "0".

layerset <- $layer_{set}$("0")

layerfreeze2 - $Layer_{freeze}$ locks the layer "echannel" so that it cannot be edited.

layerfreeze2 <- $layer_{freeze}$("echannel")

- x : $origin_0$
- y : $origin_1$
- z : $origin_2$

$platform_{dim}$ =

- x : (2(2*R) + (3*disp)
- y : (2R) + walkway + disp
- z : thick

H_{barrel} = H

$barrel1_{origin}$ =

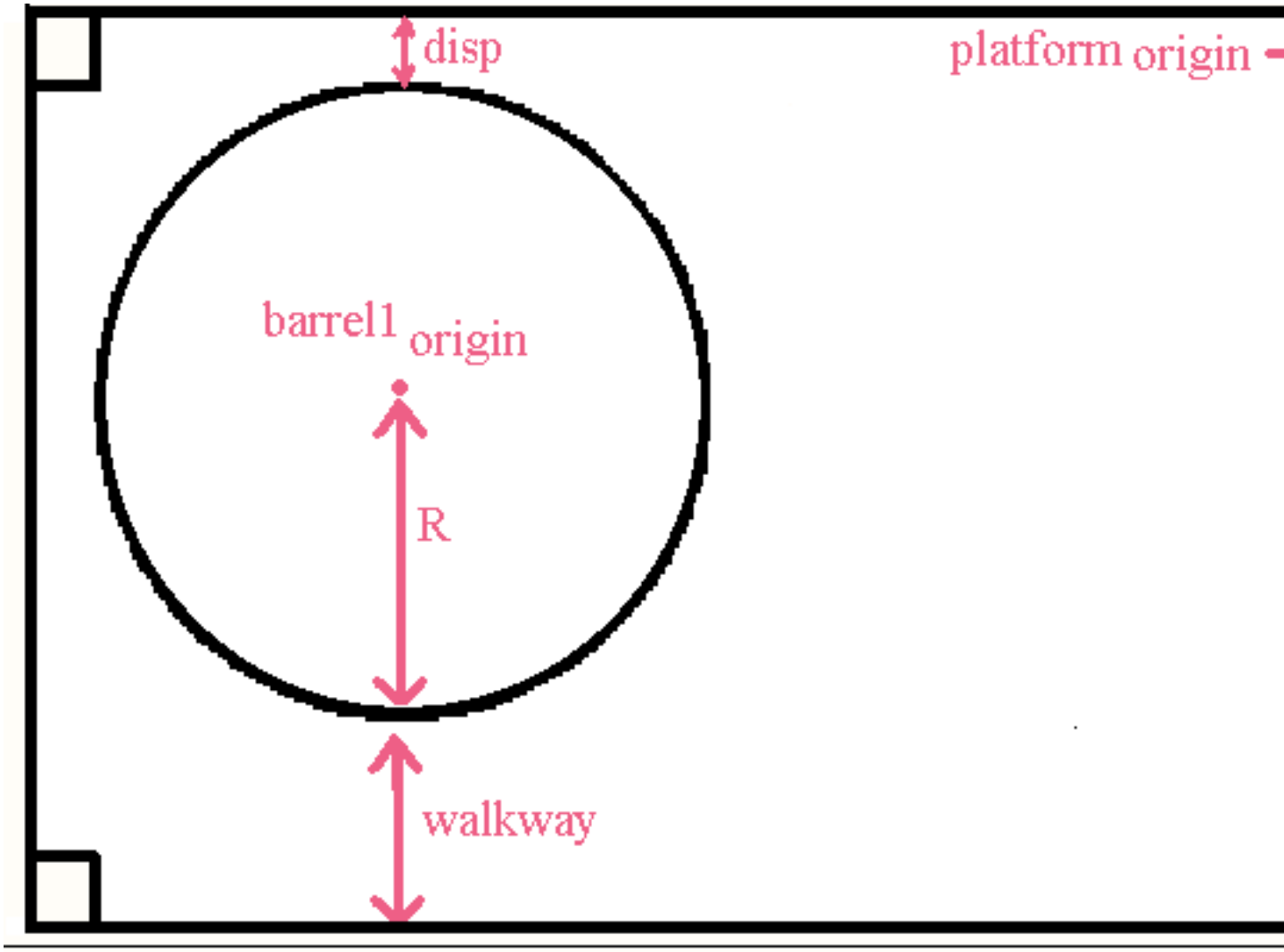
- x : $platform_{origin0}$ - ((3*R) + (2*disp))
- y : $platform_{origin1}$ - (R + disp)
- z : $platform_{origin2}$ + $platform_{dim2}$

$barrel2_{origin}$ =

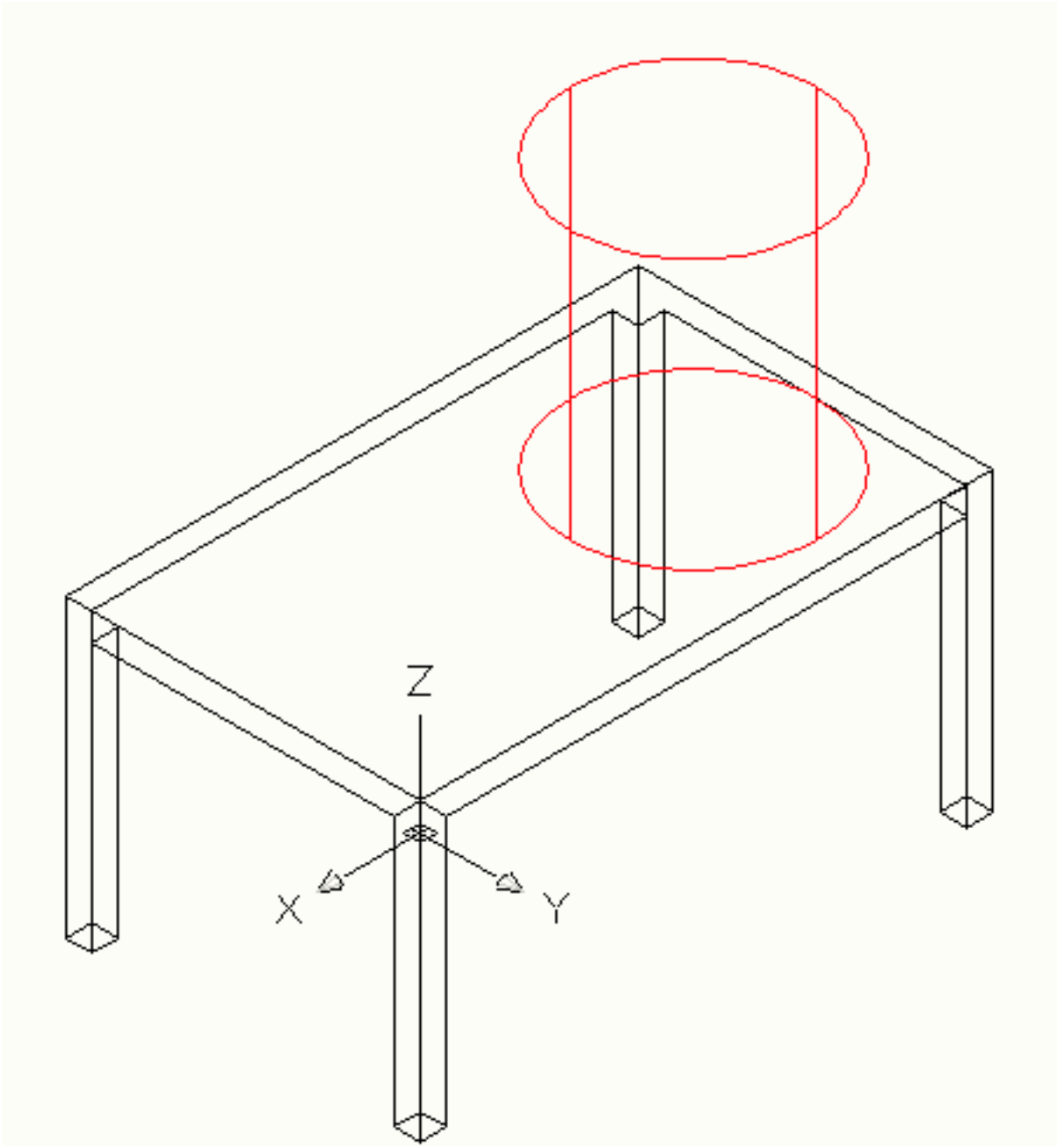
- x: $platform_{origin0}$ - (R + disp)
- y : $platform_{origin1}$ - (R + disp)
- z : $platform_{origin2}$ + $platform_{dim2}$

Technical Program Outline

cylinder1 - Creates a cylinder with the CylinderC function.



Topview



NorthEast Isometric view

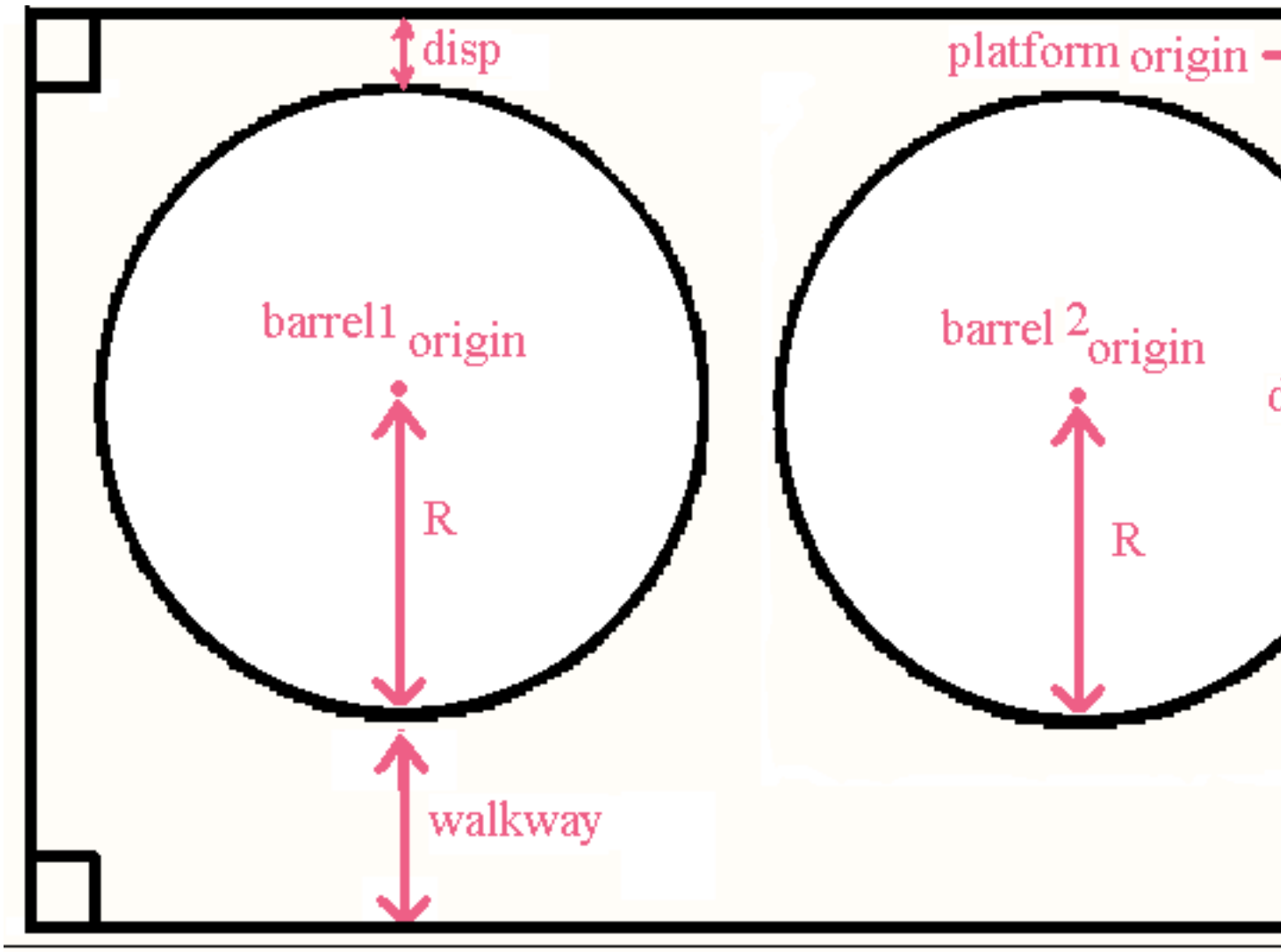
cylinder1<-cylinderC(barrel1$_{origin}$,R,H$_{barrel}$)

barrel1$_{origin}$ =

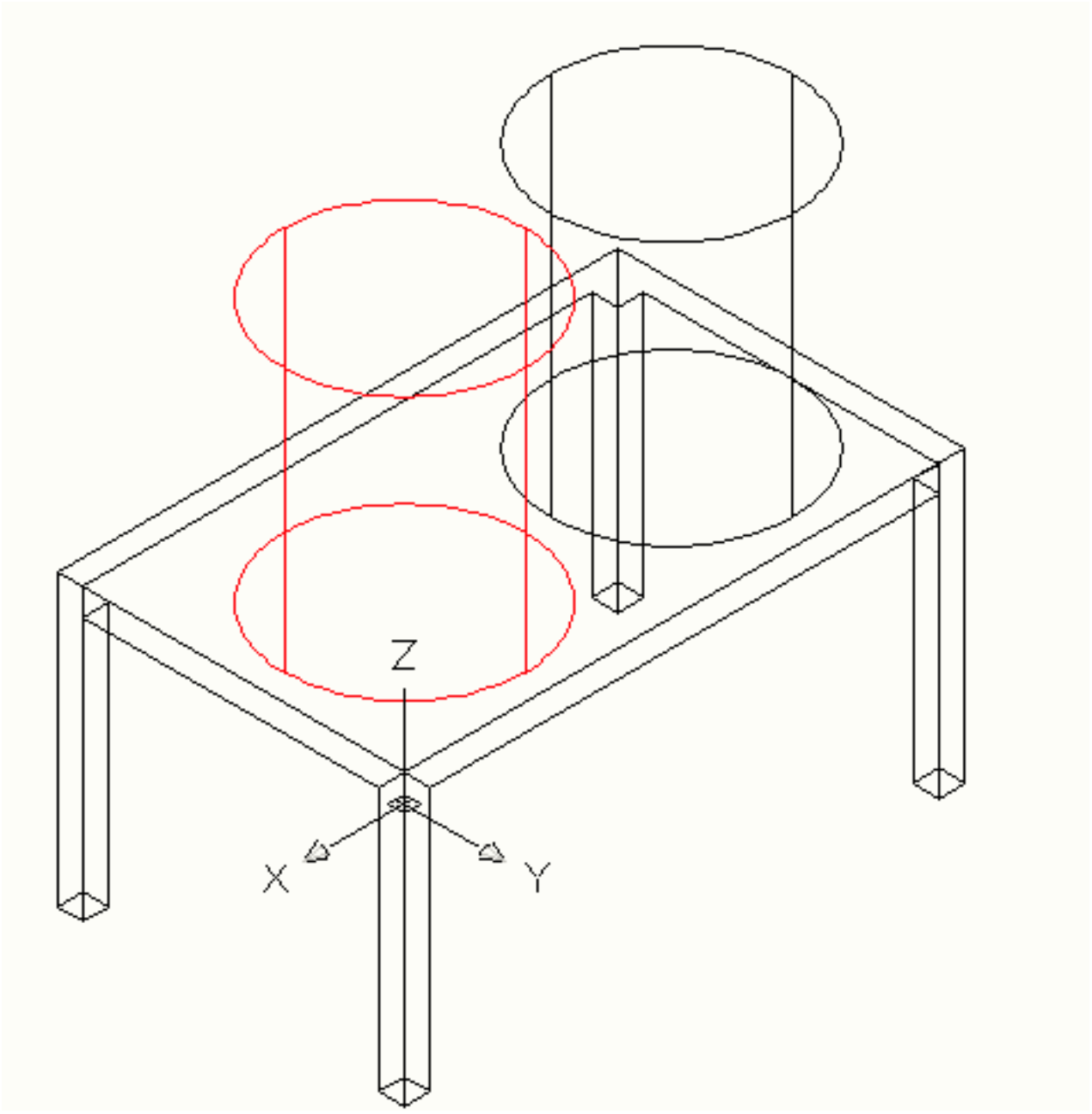
- x : platform$_{origin0}$ - ((3*R) + (2*disp))
- y : platform$_{origin1}$ - (R + disp)
- z : platform$_{origin2}$ + platform$_{dim2}$

H$_{barrel}$ = H

cylinder2 - Creates a cylinder with the CylinderC function.


disp
platform origin
barrel1 origin
R
barrel 2 origin
R
walkway

Topview



NorthEast Isometric view

cylinder2<-cylinderC($barrel2_{origin}$,R,H_{barrel})

$barrel2_{origin}$ =

- x: $platform_{origin0}$ - (R + disp)
- y : $platform_{origin1}$ - (R + disp)
- z : $platform_{origin2}$ + $platform_{dim2}$

H_{barrel} = H