For Each pt As point3d In input_pt
    Dim output_pts As New List(Of Point3d)
    output_pts.Add(pt)
    Dim output pt As point3d
    Do
        outputpoint(base_srf, pt, distance_factor, output pt)
        output_pts.add(output_pt)
        pt = output pt
        If output_pts.Count > 100 Then
            Exit Do
        End If
    Loop While outputpoint(base_srf, pt, distance_factor, output_pt) = True
    Dim output_crva As New PolylineCurve(output pts)
    output_crvs.Add(output_crva)
Next:
A = output_crvs

End Sub

Private Function outputpoint(ByVal base_srf As Surface, ByVal input_pt As Point3d, ByVal distance_factor As Double, ByRef A As Object) As Boolean
    Dim u, v As Double
    base_srf.ClosestPoint(input_pt, u, v)
    Dim normal_vector As New Vector3d(base_srf.NormalAt(u, v))
    Dim translation_vector As Vector3d = Vector3d.CrossProduct(normal_vector, vector3d.ZAxis)
    translation_vector.Unitize
    translation_vector.Transform(Transform.Rotation(Math.PI * 0.5, normal_vector, input_pt))
    Dim moved_pt As point3d = input_pt + distance_factor * translation_vector
    base_srf.ClosestPoint(moved_pt, u, v)
    Dim output pt As Point3d = base_srf.PointAt(u, v)
Session 03 : VB Scripting
Monday, August 6, 2018  2:59 PM

Visual Basic & APIs in Rhino

- Visual basic is a programming language that deals with windows API
- APIs in Rhino
  - RhinoCommon
  - Grasshopper
  - Rhino Script Syntax (Python)
  - Rhino Script

Visual Basic

- Variables
- Declaration - Class, Object, Instance
- Statements
- Loops
- Array
- Procedures - Sub vs Function

RhinoCommon API

- Namespaces
- Classes
- Constructors
- Properties
- Methods

Grasshopper API

- A collection of classes that doesn’t exist in native Rhino API

Communicating with Rhino Objects in VB

- doc. / rhino.rhinodoc.activedoc

Examples

- Converting 1d array to data tree in Grasshopper VB
- Simple component reacting surface curvature
- Drainage simulation