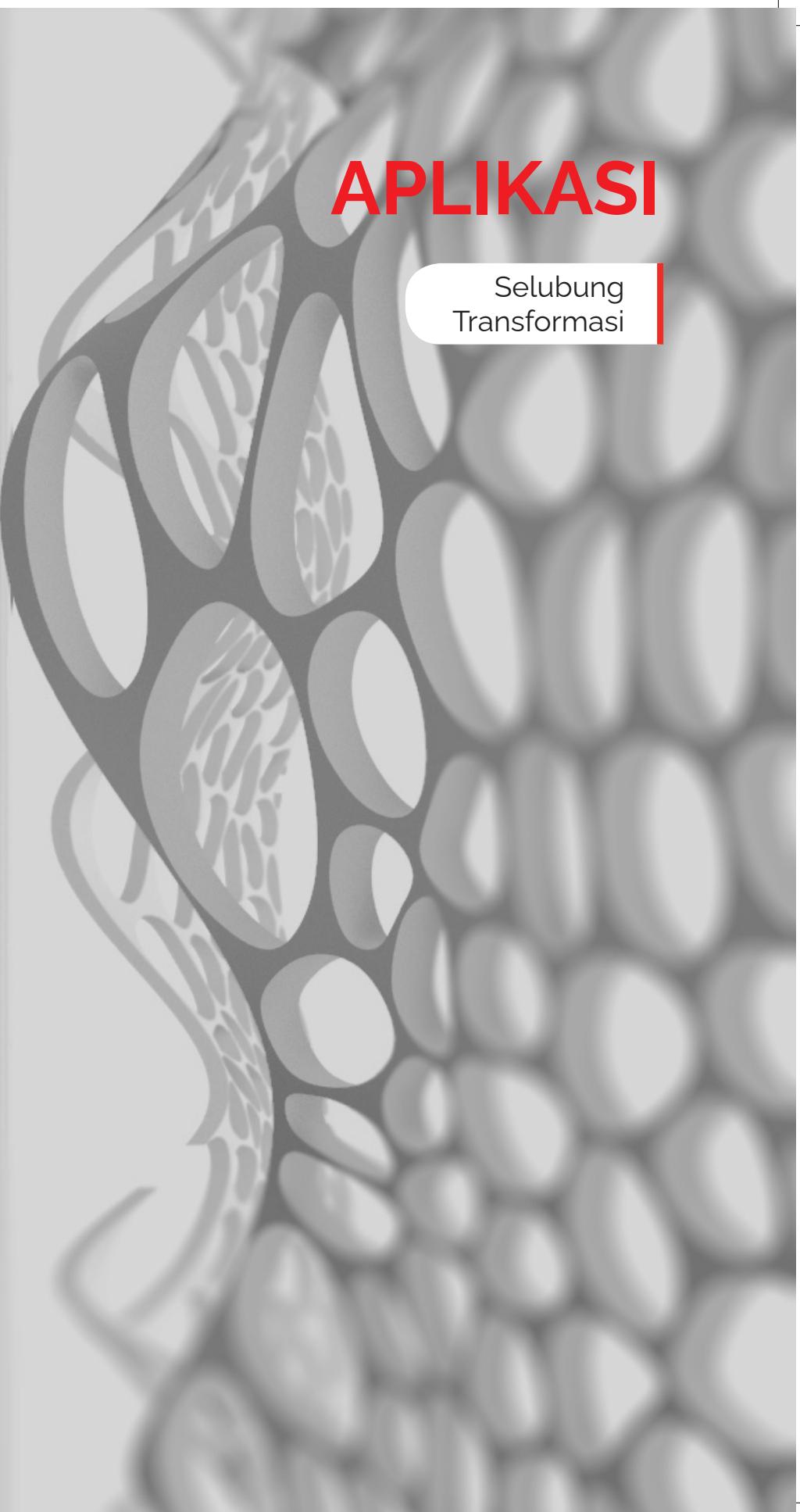


APLIKASI

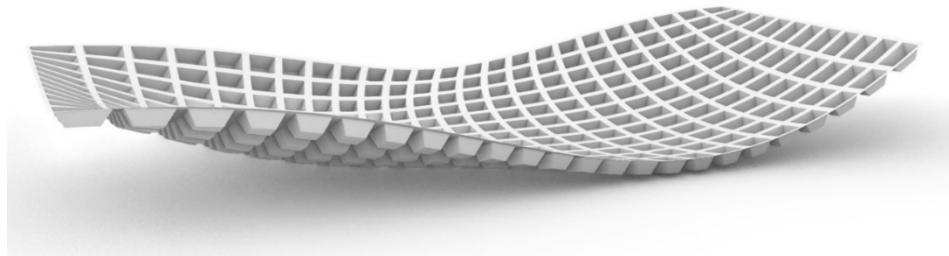
Selubung
Transformasi



7. SELUBUNG DAN FAÇADE (SKINS)

File latihan di bab ini dapat diunduh di:
<https://github.com/aswinindra/eskacangmerah>

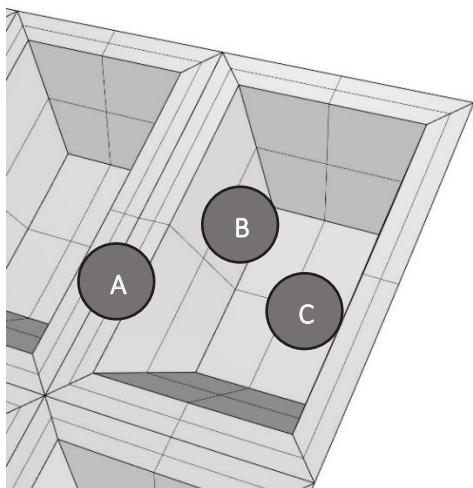
Aplikasi pemahaman atas algoritma dan data pada selubung (*skin*) bangunan dapat digunakan untuk merancang pola-pola baik 2D atau 3D pada bidang-bidang baik planar maupun non-planar misalnya pada fasad, atap atau elemen-elemen lain.



Gambar 164. Desain Pola Wafel 3D Pada Permukaan Non-Planar

7.1. Grid Wafel (Rectangular Pattern)

Kita akan membuat pola 3D menggunakan sebuah obyek surface sebagai referensi. Hasil final seperti terlihat di gambar di atas adalah pola pyramid (**waffle grid**) atau dinamakan *pyramidal frustum*. Bagaimana algoritmanya?



Pola 3D dibuat dengan memanipulasi grid pada surface referensi. Ada tiga surface yang akan dihasilkan dalam prosesnya: Surface A, Surface B dan Surface C.

Surface A adalah surface referensi yang dibagi menjadi grid.

Surface B adalah surface yang membuat pola 3D melalui elemen grid yang dinaikkan/diturunkan.

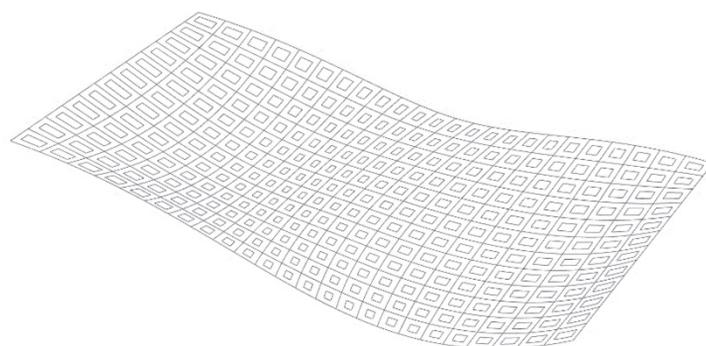
Surface C adalah surface yang menutup surface atau pola 3D.

Gambar 165. Tiga Bidang Surface

Prosedur atau algoritma:

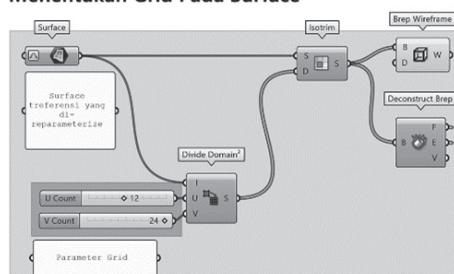
1. Surface referensi di-reparameterize. Bagi menjadi grid dan ekstrak grid menjadi *Face*, *Edge* dan *Vertices*.
2. Offset edges (gunakan komponen **Offset on Surface** karena bidang offset tidak planar), buat surface antara edge asal dan edge hasil offset (Surface A).
3. Pindahkan edge hasil offset ke atas/ ke bawah sesuai dengan *vector normal* dari setiap grid, gunakan **Edge Surface** untuk membuat permukaan 3D (Surface B).
4. Gunakan **Edge Surface** untuk membuat surface pada edge yang telah dipindahkan (Surface C).

Langkah 1:Membagi Surface

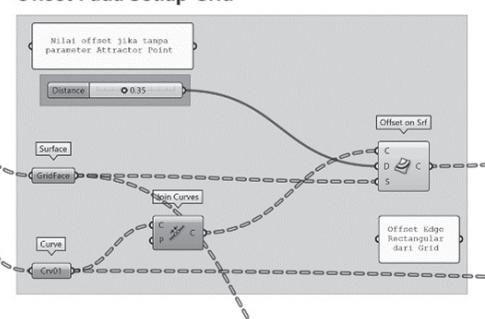


Ar-4214 TAHUN 2021

Menentukan Grid Pada Surface

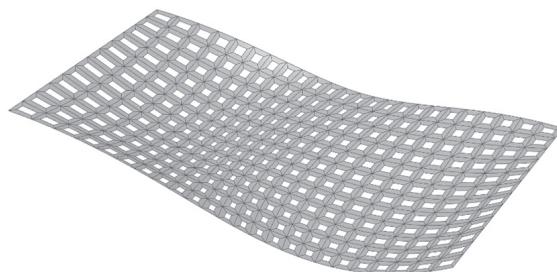


Offset Pada Setiap Grid

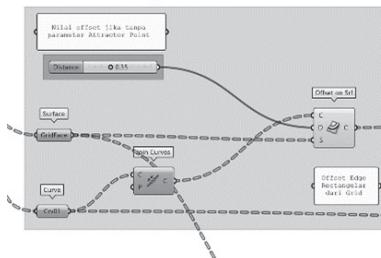


- **Reparameterize Surface**, bagi menjadi grid dengan parameter u dan v, ekstrak hasil surface dengan **Deconstruct Brep**.
- Namakan **GridFace** untuk hasil F dengan komponen **Surface**, namakan **Curve01** untuk hasil E dengan komponen **Curve**.
- Offset komponen **Curve01** dengan komponen **Offset on Surface**, buat parameter jarak offset.
- *Pertanyaan untuk dijawab. Mengapa menggunakan komponen **Join Curve** untuk komponen **Curve01** sebelum dimasukkan ke komponen **Offset on Surface**?*

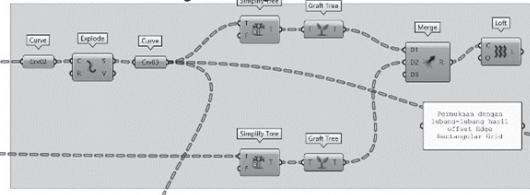
Langkah 2: Membuat Beberapa Surface



Offset Pada Setiap Grid

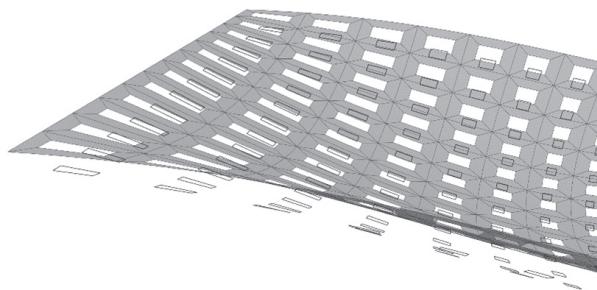


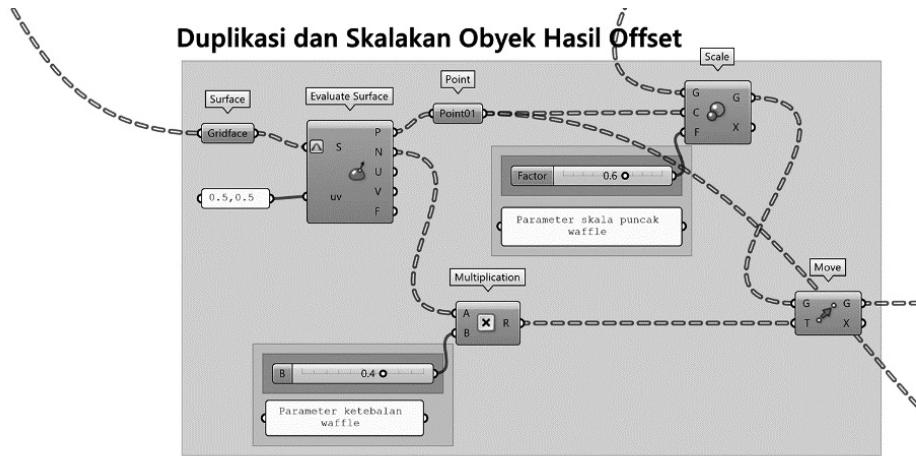
Membuat Surface Dengan Lubang Offset



- Namakan kurva hasil Offset on Surface sebagai Curve02 (rectangular yang berada di dalam kotak grid), explode Curve02, namai hasilnya sebagai Curve03. Anda perhatikan, kita memiliki dua buah kumpulan kurva: Curve01 dan Curve03 yang masing-masing sudah berupa segmen.
- Pada prinsipnya data dari masing-masing kumpulan segmen ini (Curve01 dan Curve03) di-loft dengan komponen Loft untuk membuat Surface. Tapi anda harus perhatikan struktur datanya. Masing-masing memiliki **cabang yang berbeda**, sehingga harus dipangkas, dijadikan satu cabang untuk masing-masing data menggunakan komponen Simplify Tree. Sekarang masing-masing cabang sudah memiliki empat segmen. Agar komponen Loft bisa bekerja sesuai dengan yang diharapkan (index 0 pada obyek A bertemu dengan index 0 pada obyek B, dan seterusnya), maka setiap data ini harus di-graft menggunakan komponen Graft Tree. Anda dapat lihat, menggunakan komponen Merge, obyek-obyek ini dapat di-loft.

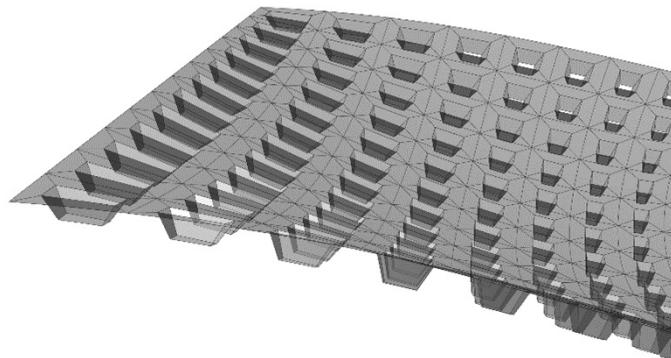
Langkah 3: Menentukan Kurva Panduan untuk Loft

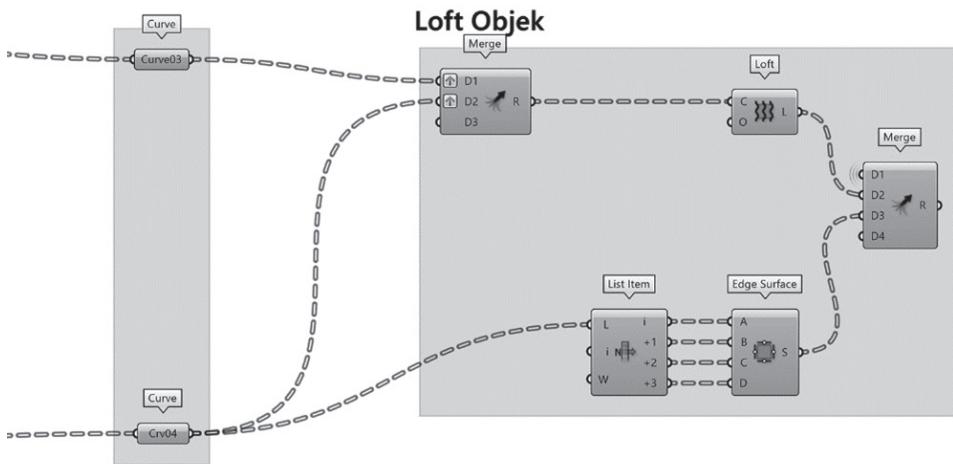




- Masukkan komponen **Evaluate Surface** dengan input **Gridface** untuk menentukan titik tengah dari setiap gridface. Tentukan uv parameter dengan nilai 0.5,0.5.
- Masukkan komponen **Scale** dan tentukan C (center) pada output P (Point) dari komponen **Evaluate Surface** dengan nilai Faktor yang dapat anda tentukan secara parametrik. Nilai ini adalah nilai offset untuk menghasilkan obyek hijau di atas, sehingga nilai Scale harus <1.
- Pertanyaan untuk dijawab. Komponen apakah yang dihubungkan pada input G (geometry) pada komponen Scale ini?*
- Masukkan komponen **Move** untuk memindahkan obyek hasil scale. Nilai Faktor dapat anda tentukan secara parametrik. Nilai (+) akan searah dengan vector normal (n) dan nilai (-) akan berlawanan dengan vector normal. Ini adalah parameter ketebalan atau ketinggian wafel atau pyramid.

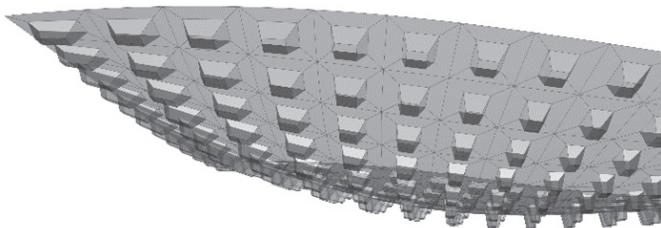
Langkah 4: Membuat Loft Surface





- Beri nama Curve04 dari hasil dari komponen Move dan dengan menggunakan komponen Merge, buat permukaan antara Curve3 dan Curve4 menggunakan komponen Loft. Ingat anda harus melakukan Graft pada masing-masing data Curve tersebut agar komponen Loft bekerja sesuai harapan. (anda bisa gunakan komponen Graft Tree atau parameter Graft pada input parameter pada komponen Merge).

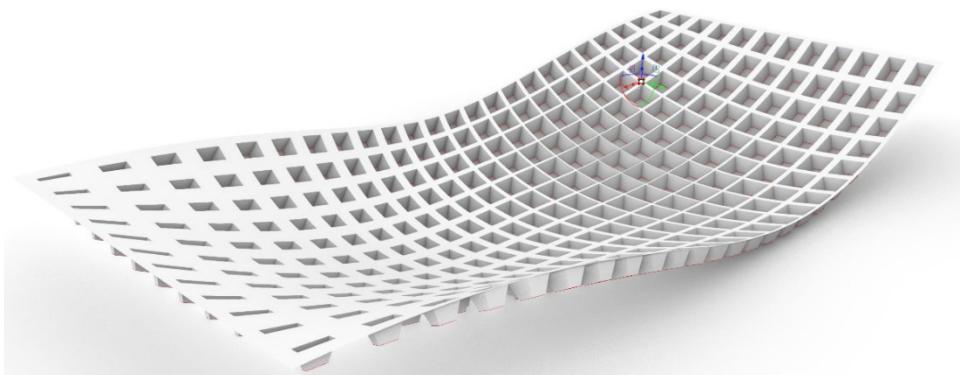
Langkah 5: Membuat Edge Surface



- Terakhir, buat surface dari Curve04 menggunakan Edge Surface dimana setiap item dari Curve04 diekstrak menggunakan List Item.
- Ada dapat mengumpulkan ketiga obyek surface melalui komponen Merge.

Latihan 12: Pola Wafel

1. Kembangkan definisi sebelumnya sehingga pola wafel yang dihasilkan responsif terhadap jarak ke suatu titik.

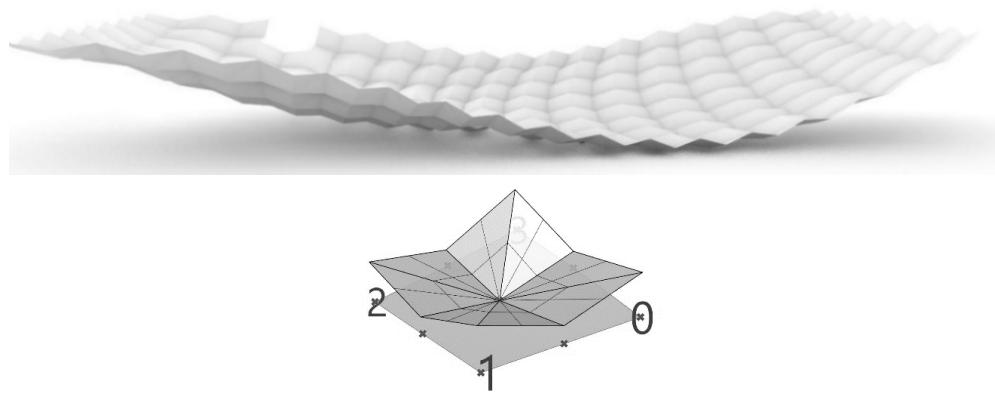


Gambar 166. Pola Grid Wafel 3D Responsif Terhadap Titik Attraktor

7.2. Modifikasi Elemen *Gridface*

Pengembangan dari contoh bagian pertama adalah modifikasi dari elemen-elemen pada suatu grid. Anda sudah paham bagaimana grid diproduksi, bagaimana mengekstrak tiga komponen dari masing-masing grid yakni: *Gridfaces*, *Edges* dan *Vertices* dan bagaimana memanfaatkan informasi melalui modifikasi struktur data untuk membuat obyek berbasis grid.

Selanjutnya, pengembangan pemahaman atas komponen-komponen *Gridface* dapat digunakan untuk aplikasi panel dengan contoh berikut.



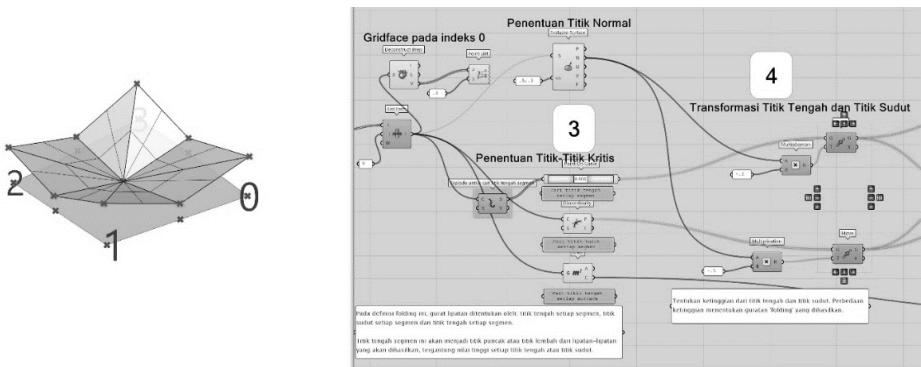
Gambar 167. Modifikasi Elemen *Gridface*

Prosedur dan algoritma:

1. Untuk memudahkan proses pengerjaan algoritma, ekstrak satu gridface hasil dari pembagian grid pada kurva referensi.

- Bentuk yang akan dibuat menyerupai lipatan/folding dimana setiap titik sudut grid dipindahkan ke atas, setiap titik tengah Edge juga dipindahkan ke atas dengan jarak berbeda dengan jarak titik-titik Edge. Perhatikan bahwa pemindahan ini bukan ke sumbu Z tetapi ke vector normal gridface tersebut. Semua titik ini dihubungkan ke titik pisat Gridface.
- Ekstrak titik-titik pada Edges dan titik tengah grid, kemudian lakukan transformasi pada titik-titik di Edge sesuai dengan ketinggian permukaan.
- Hubungkan titik-titik hasil perpindahan tadi ke titik tengah Gridface dan buat permukaan.
- Setelah yakin dengan hasilnya, maka anda bisa terapkan untuk seluruh Surface.

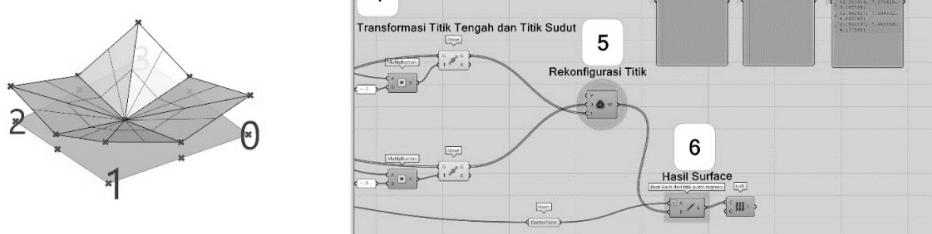
Langkah 1: Ekstrak Titik-Titik



BUKU AJAR AR-4214 TAHUN 2021

- Penentuan titik tengah suatu segmen menggunakan komponen **Point on Curve**, penentuan titik-titik pojok pertemuan antar segmen menggunakan komponen **Discontinuity**, penentuan titik tengah suatu surface menggunakan komponen **Area**.
- Gunakan komponen **Evaluate Surface** untuk menentukan titik tengah suatu surface sekaligus vector normalnya.
- Gunakan vector normal sebagai nilai untuk translasi titik-titik menggunakan komponen **Move. Ingat**, anda harus cek arah transformasinya, di atas atau di bawah surface. Gunakan (-) atau (+) sebagai faktor pengali untuk translasi ini.

Langkah 2: Translasi Titik-Titik



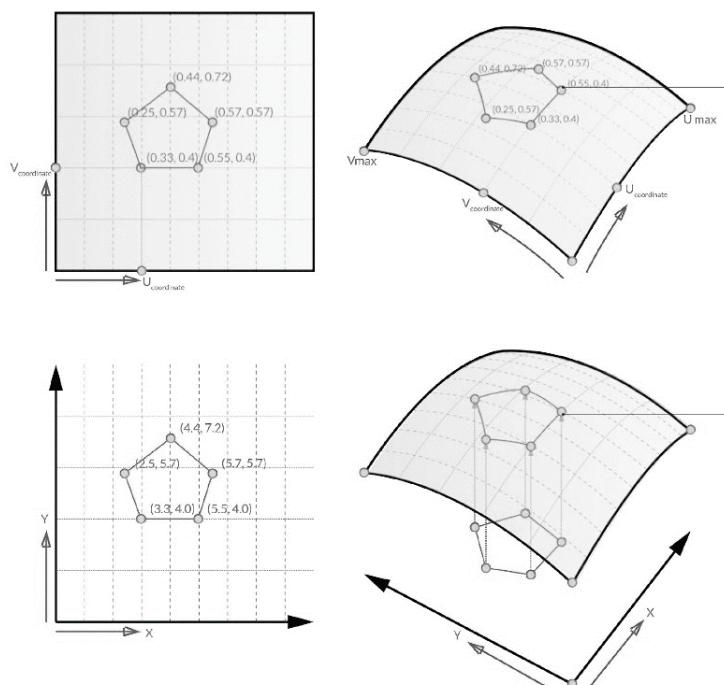
- Titik-titik pojok dan titik-titik tengah tiap edge harus dijadikan satu namun lis-nya dibuat selang-seling (weave) menggunakan komponen Weave.
- Komponen Line digunakan untuk membuat garis antara setiap titik-titik di atas dengan titik pusat GridFace.
- Komponen Loft digunakan untuk membuat Surface.

Langkah 3: Aplikasi Pada Seluruh Surface

- Tugas: aplikasikan algoritma ini pada keseluruhan Surface.

7.3. Projection, Mapping & Morphing

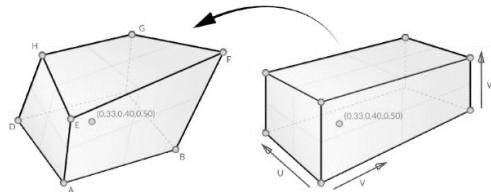
Berdasarkan konsep NURBS Surface yang sudah disampaikan di bab sebelumnya bahwa setiap Surface memiliki sistem koordinat sendiri (LCS- Local Coordinate System, yang memiliki aksis u, v dan w). Dengan demikian obyek yang berada di luar Surface tersebut dengan koordinat WCS (x,y,z) dapat di-mapping pada sistem koordinat u,v,w pada permukaan Surface (ingat komponen **Curve Closest Point**). Geometri yang di-mapping ini akan mengalami deformasi sebagai respon terhadap profil kelengkungan dari surface tersebut. Di sini kita akan memahami perbedaan antara konsep *Mapping* dan *Projection*. *Mapping* adalah memetakan obyek 2D atau 3D ke suatu permukaan dimana setiap titik-titik obyek pada koordinat WCS akan ditransformasi ke koordinat local (LCS). Sedangkan *Projection* adalah menarik garis proyeksi setiap titik pada obyek sehingga berpotongan dengan permukaan proyeksi.



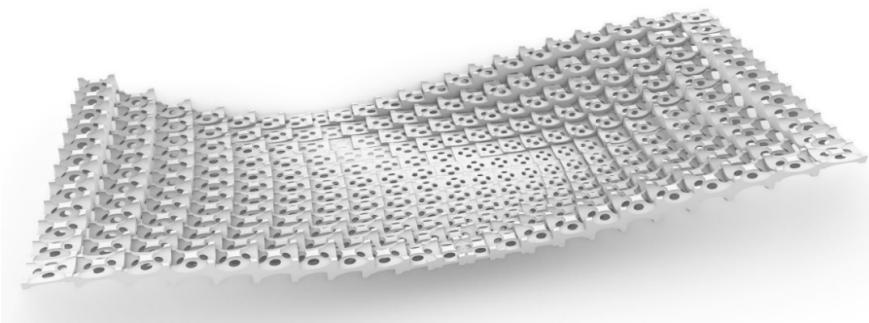
Gambar 168. Mapping & Projection. (Sumber: Gil Akos & Ronie Parsons, 2017)

Seperti halnya obyek 2D, obyek 3D ketika di-*mapping* ke suatu Surface akan mengalami deformasi/*morphing*.

Ketika dipetakan/*mapping* pada suatu permukaan, setiap obyek 3D akan terdeformasi/*morphing* sesuai dengan kelengkungan permukaan tersebut. Deformasi ini tidak mengurangi derajat keutuhan/*topology* obyek 3D tersebut.

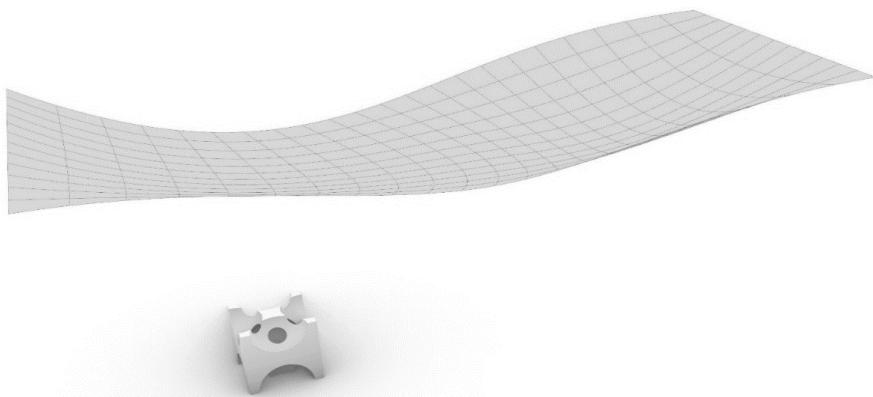


Gambar 169. Mapping & Topology. (Sumber: Gil Akos & Ronie Parsons, 2017)



Gambar 170. BoxMorph

Obyek di atas adalah contoh penerapan morphing obyek 3D pada sebuah permukaan/Surface.

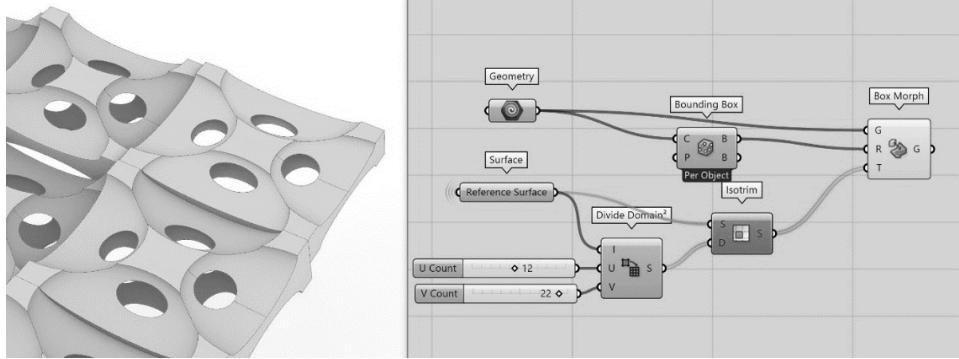


Gambar 171. Surface dan Elemen 3D Untuk BoxMorph

Prosedur dan algoritma:

1. Buat obyek 3D yang akan dipetakan/di-*morph* pada permukaan referensi. Perhatikan ukuran obyek 3D disesuaikan dengan besar grid pada permukaan.

2. Bagi Surface menjadi grid dengan parameter u dan v.
3. Definisikan obyek 3D sebagai obyek yang akan di-*morph* menggunakan komponen Bounding Box.
4. Gunakan komponen Box Morph untuk memetakan obyek 3D, dengan referensi dari Bounding Box, pada Surface.



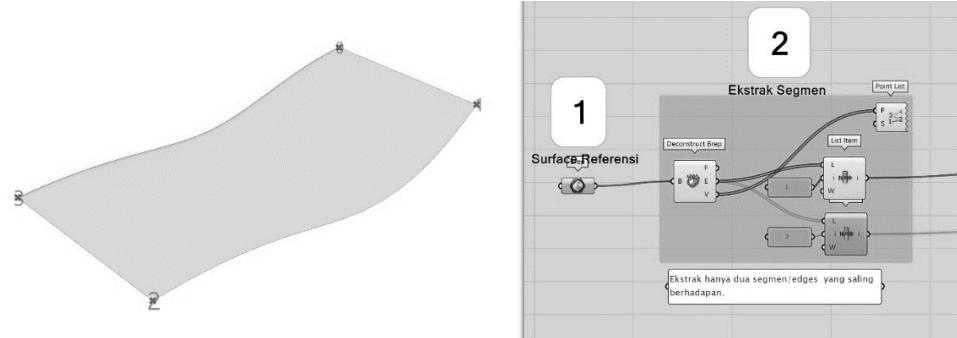
7.4. Twisting Louvers

Implementasi lainnya dari pemahaman ekstraksi komponen *surface* dan struktur data adalah membuat sirip-sirip pada suatu permukaan yang diputar pada salah satu aksisnya.

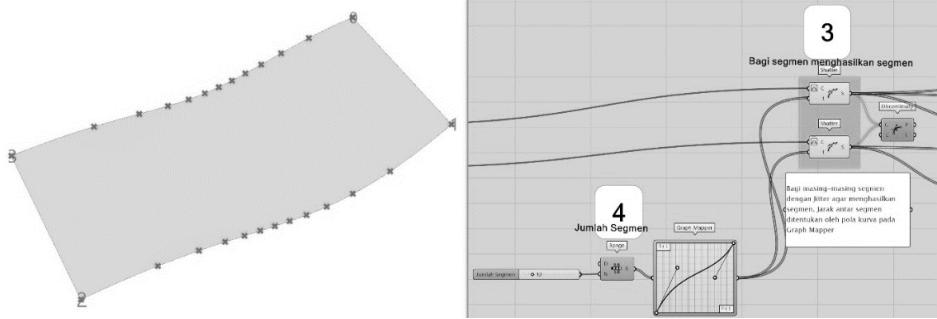


Gambar 172. Twisting Louvers

Langkah 1: Membagi segmen pada masing-masing Edge

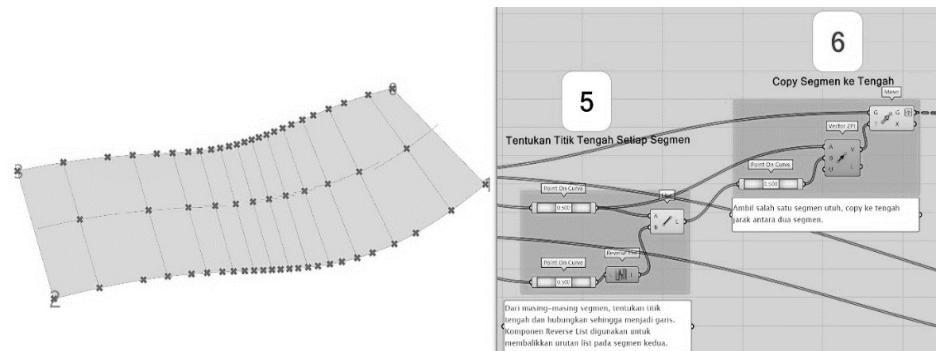


- Ekstrak Surface dengan komponen Deconstruct Brep, ekstrak hanya dua Edge yang saling berhadapan menggunakan List Item.



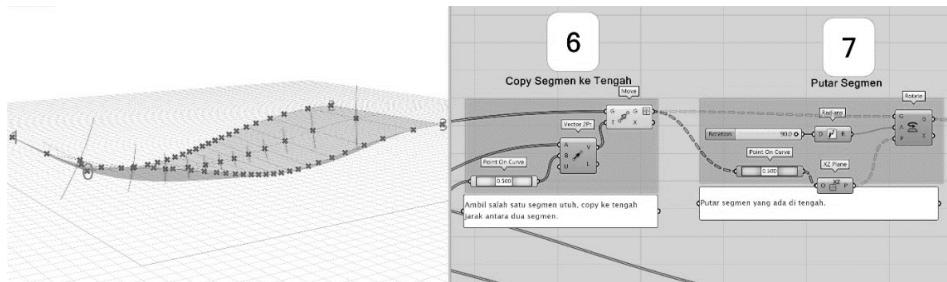
- Kita membagi masing-masing Edge agar menghasilkan segmen yang sama antara Edge satu dan Edge lainnya. Gunakan komponen Shatter untuk membagi segmen menghasilkan segmen dengan nilai t dari Graph Mapper. Kita dapat membuat pola jarak antar segmen sesuai dengan yang kita inginkan.
- Pertanyaan untuk dijawab. Kalau kita menginginkan jarak antar segmen sama, bagaimana pola pada Graph Mapper?
- Anda bisa cek titik-titik endpoints dari setiap segmen menggunakan komponen Discontinue.

Langkah 2: Membuat segmen referensi pada tengah surface



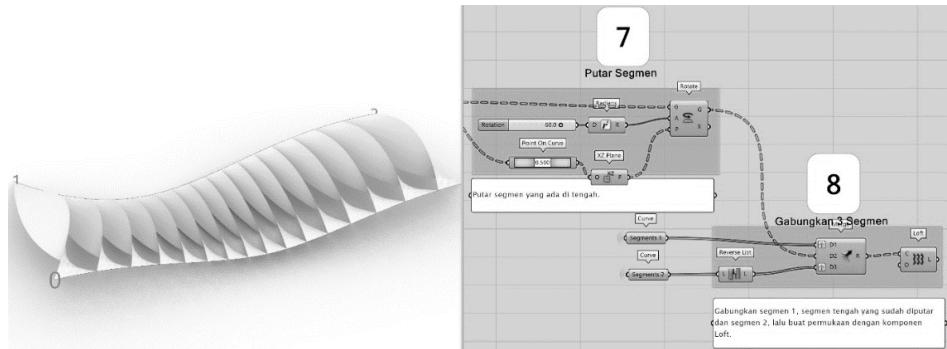
- Tentukan titik tengah masing-masing segmen dengan komponen Point on Curve=0.5, buat garis yang menghubungkan kedua titik, jangan lupa salah satu lis titik tengah harus dibalik nomor indeksnya menggunakan komponen Reverse List (*mengapa? Anda bisa coba kalau tidak dibalik*).
- Buat garis antara titik-titik itu dengan komponen Line.
- Pindahkan/copy salah satu segmen garis di salah satu Edge ke tengah menggunakan komponen Move, dengan nilai factor menggunakan komponen Vector 2Points. Apa fungsi komponen ini? Penjelasannya demikian: translasi obyek memerlukan arah vector, dan Vector 2Pt berfungsi memberikan informasi arah itu dengan dua titik. Dalam konteks ini dua titik itu adalah: 1) titik tengah antar segmen; 2) titik tengah garis yang baru dibuat.

Langkah 3: Rotasi segmen tengah



- Segmen yang dipindahkan ke tengah (berada pada bidang XY) akan dirotasikan menggunakan komponen **Rotate**. Komponen ini memerlukan input berupa: obyek yang akan dirotasi, sudut (dalam radian) dan sumbu putar.
- Obyek yang diputar adalah segmen yang sudah dipindah ke tengah, nilai derajat dimasukkan dan masukkan komponen **Degree2Radian** untuk konversi nilai sudut putar.
- Input P (Plane) pada komponen Rotate memerlukan posisi dan bidang aksis. Posisi aksis berada di tengah segmen (anda gunakan komponen Point on Curve=0.5) pada obyek segmen yang sudah dipindah ke tengah dan sumbu aksis menggunakan komponen XZ Plane. Mengapa XZ Plane? Karena obyek akan kita putar pada bidang XZ (perhatikan cumbu X adalah garis merah pada Rhinoceros).
- Hasil akhir semua segmen di tengah garis berputar pada nilai sudut putar pada sumbu XZ di tengah garis.

Langkah 4: Gabungkan segmen

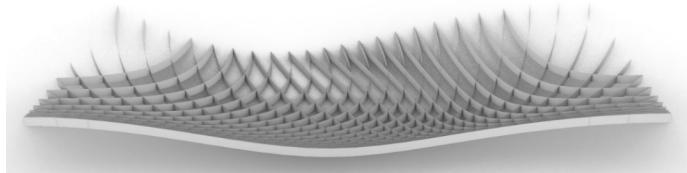


- Gabungkan segmen pada masing-masing Edge dengan segmen yang baru diputar, gunakan komponen **Loft** untuk membuat permukaan.

Teknik ini dapat anda kembangkan untuk membuat obyek-obyek fin, ribs dengan parameter-parameter baru yang bisa diambahkan.

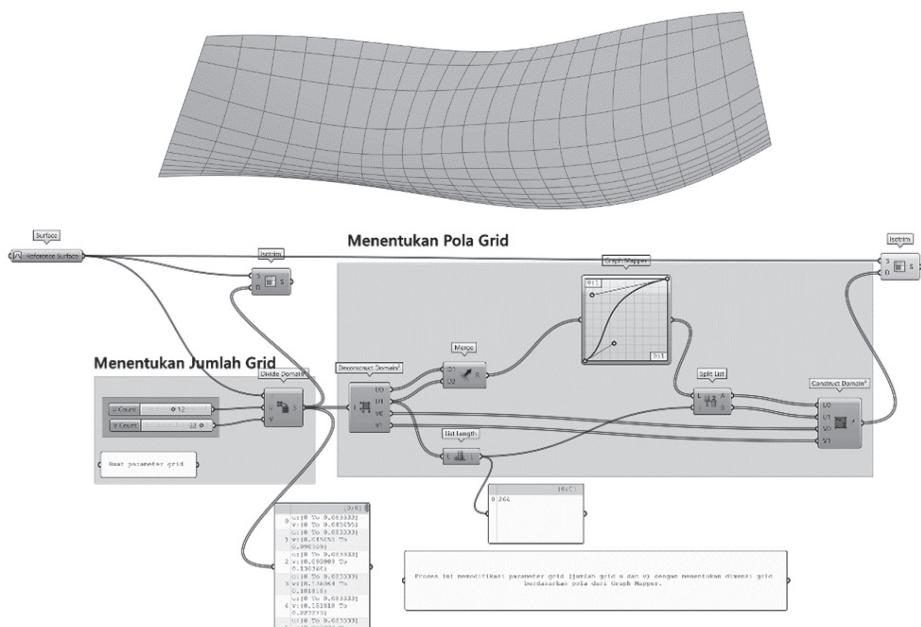
7.5. Interpolasi Diagrid (*Lattice of Intersecting Fin*)

Contoh berikut adalah implementasi dari modifikasi struktur data melalui re-organisasi, interpolasi pada titik-titik grid, menghasilkan pola diagrid dengan struktur grid parametrik dan interpolasi garis yang melalui titik-titik grid. Contoh ini juga menunjukkan bagaimana membuat grid pada Surface yang berpola.



Gambar 173. Pola Lattice Pada Diagrid

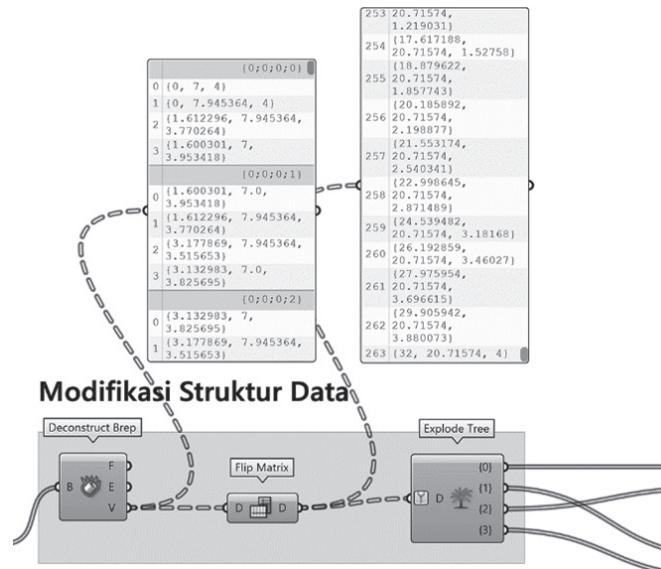
Langkah 1: Menentukan Pola Grid



- Pastikan Reference Surface sudah di-reparameterized.
- Komponen **Divide Domain** membuat format grid dalam parameter *u* dan *v*. Ekstrak masing-masing domain ini menggunakan komponen **Deconstruct Domain**² (Perhatikan ada dua komponen dengan nama yang sama). Komponen ini akan mengekstrak empat keluaran: *u0(min)*, *u1(max)*, *v0(min)* dan *v1(max)*. Ini adalah dua domain yang akan kita modifikasi.
- Pada dua domain *u* (*u0* dan *u1*), gunakan komponen **Merge** dan masukkan pada komponen **Graph Mapper**. Tentukan pola grafik yang diinginkan. Pada bagian ini kita akan memetakan domain *u* (*u0* dan *u1*), artinya posisi jarak-jarak Isocurve-nya sesuai dengan pola grafik.

- Tentukan Panjang list dari domain u menggunakan **List Length**.
- Gunakan komponen **Split List** untuk membagi list antara hasil **Graph Mapper** (u_0) dengan hasil dari **List Length**.
- Masukkan komponen **Construct Domain²**, dan gunakan input-input sesuai gambar di atas. Di sini kita lihat bahwa domain u (u_0 dan u_1) sudah dimodifikasi dengan **Graph Mapper**.
- Hasil ini kita masukkan ke komponen **Isotrim** untuk membagi Surface menjadi Gridfaces.

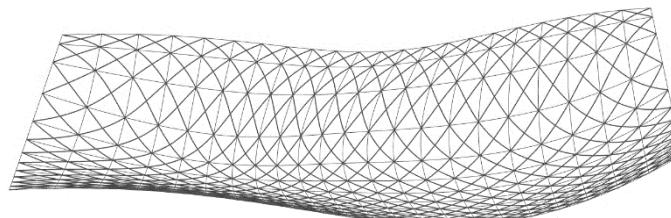
Langkah 2: Modifikasi Struktur Data

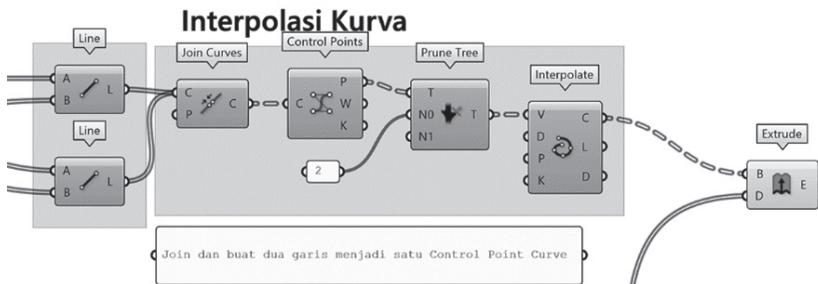
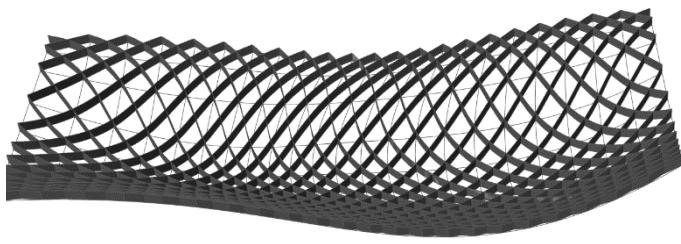


BUKAJAR AR-4214 TAHUN 2021

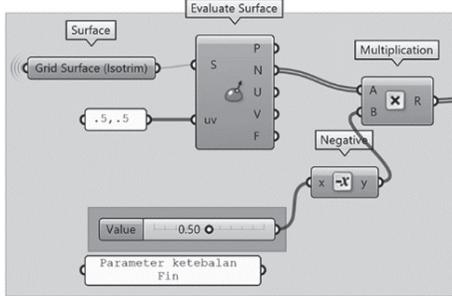
- Struktur data vertices dibalik menggunakan komponen **Flip Matrix**. Anda dapat mengecek urutan nomor index setiap vertices.
- Simplify Tree dan gunakan komponen **Exploded Tree** untuk membuat list dari masing-masing path.
- Buat garis menggunakan komponen **Line** pada empat path tersebut.

Langkah 3: Membuat Kurva Dengan Interpolasi





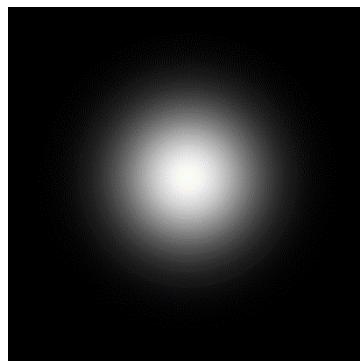
Menentukan Ketebalan



- Gabungkan dua objek garis menggunakan **Join Curve**, tentukan masing-masing titik pada garis sebagai control point menggunakan komponen **Control Points**.
- Gunakan komponen **Prune Tree** untuk menghilangkan path yang memiliki data kurang dari x (N_0). Nilai x kita masukkan sendiri, dalam contoh di atas adalah jika ada titik Control Point kurang dari 2 maka tidak akan dihitung dalam pembuatan kurva menggunakan Interpolate.
- Gunakan komponen **Interpolate** untuk membuat kurva. Anda dapat menentukan tingkat kelengkungan menggunakan parameter Knot (k).
- Untuk menentukan ketinggian atau ketebalan fn , kita gunakan vector normal dari masing-masing titik tengah pada gridface.
- Gunakan komponen **Extrude** untuk membuat ketebalan.

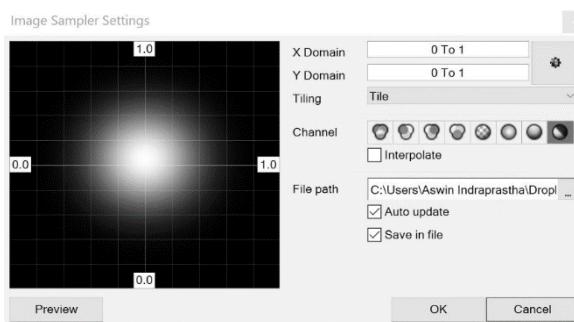
7.6. Image Sampler

Komponen **Image Sampler** berfungsi melakukan konversi nilai kromatik dari suatu Image ke nilai numerik.

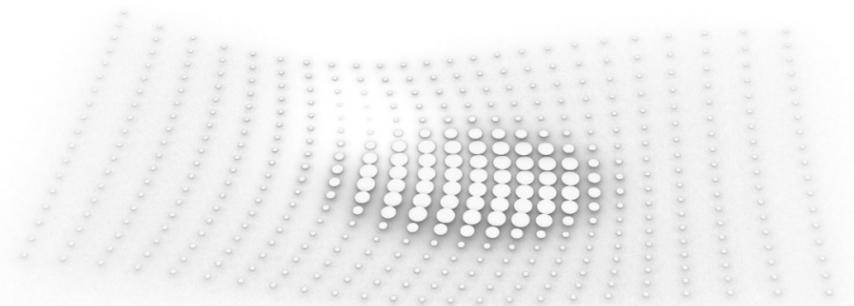


Setiap image diperlakukan sebagai domain dua dimensi yang memiliki nilai intensitas antara 0 dan 1. Nilai intensitas 1= putih, nilai intensitas 0=hitam.

Gambar 174. Contoh Raster Image



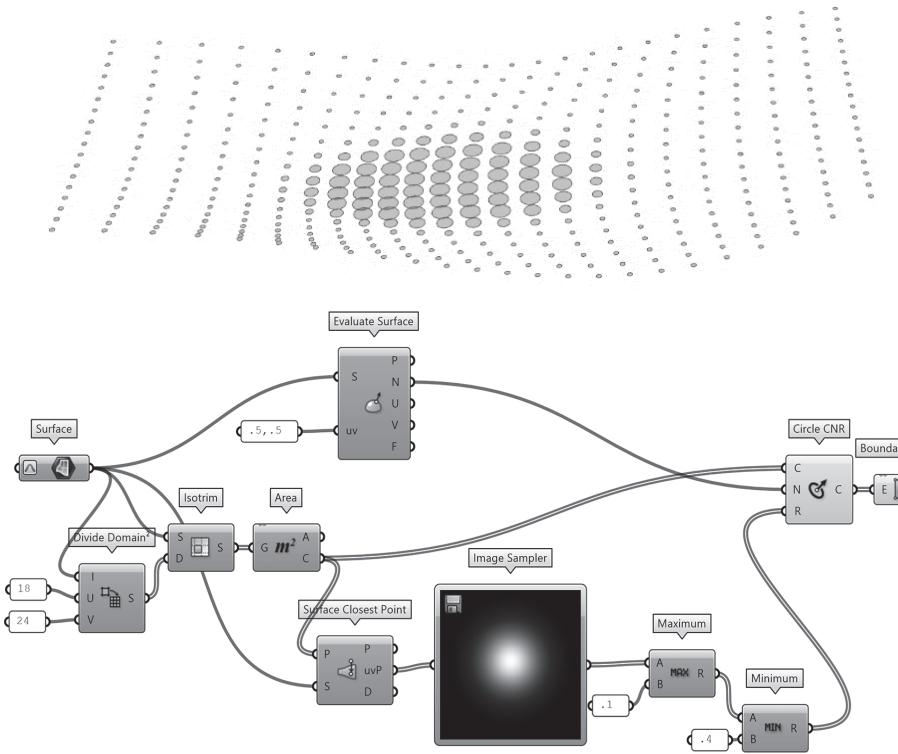
Gambar 175. Properti Raster Image di GH



Gambar 176. Transformasi Nilai Kromatik Pada Pola Radius

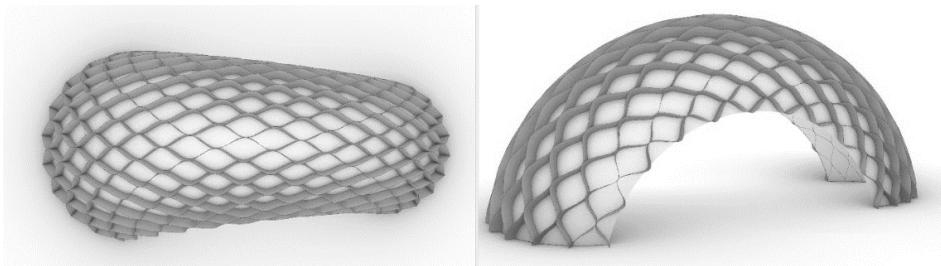
- Siapkan surface dan bagi menjadi sub surface (**Divide Domain²** & **Isotrim**). Tentukan titik pusat setiap grid dengan komponen **Area**.
- Komponen **Image Sampler** digunakan untuk menentukan besarnya radius pada setiap titik.

Langkah- Langkah



- Komponen **Image Sampler** memerlukan informasi plane setiap titik pusat di setiap grid face. Komponen yang digunakan untuk menghasilkan plane (uv) pada setiap gridface adalah **Surface Closest Point**. Komponen ini berfungsi menghasilkan titik paling dekat pada suatu surface. Jika komponen ini mendapatkan input P dari setiap titik pusat gridface, maka ia akan menghasilkan plane (uv) pada titik tersebut.
- Hasil dari komponen Image Sampler adalah nilai dari 0 sampai 1 berdasarkan nilai intensitas dari warna image bersangkutan. Nilai ini akan dipetakan pada setiap titik pusat dari gridface yang akan menentukan nilai radius komponen **Circle**.
- Untuk mengatur dan menyaring radius lingkaran, kita bisa gunakan komponen **Max** dan **Min**. Komponen Max hanya akan meloloskan nilai intensitas > 0.1 sedangkan komponen Min hanya meloloskan nilai lebih < 0.4 . Batas-batas nilai ini bisa anda tentukan sendiri.

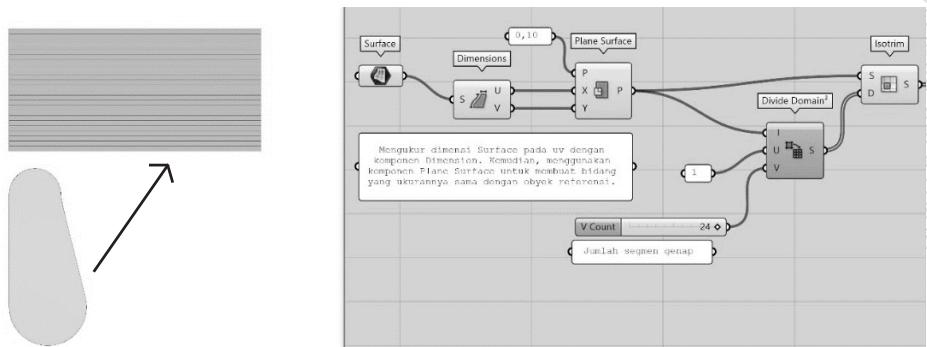
7.7. Pola Gelombang (Wavy Pattern)



Gambar 177. Pola Gelombang Pada Permukaan Non-planar

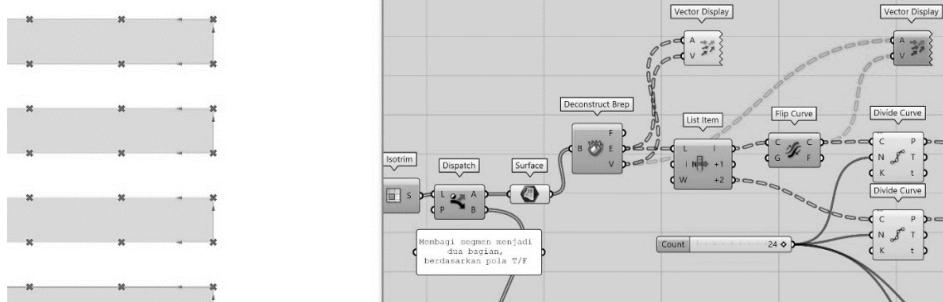
- Surface referensi didefinisikan dalam bentuk bidang dengan dimensi u dan v. Perhatikan bahwa Surface yang anda buat pada Rhino harus merupakan NURBS surface.
- Bidang dibagi menjadi strip (grid dengan salah satu sisi bernilai 1).
- Masing-masing strip dibagi menjadi beberapa segmen, masing-masing segmen dibuat sedemikian rupa sehingga jika diterapkan komponen Polyline atau Interpolate, hasilnya adalah selang-seling. Hasil ini kemudian di-extrude pada arah sumbu z.
- Hasil akhir pola wavy dipetakan/*di-mapping* ke surface referensi.

Langkah 1: Transformasi Ukuran/ Dimensi Obyek/Surface referensi



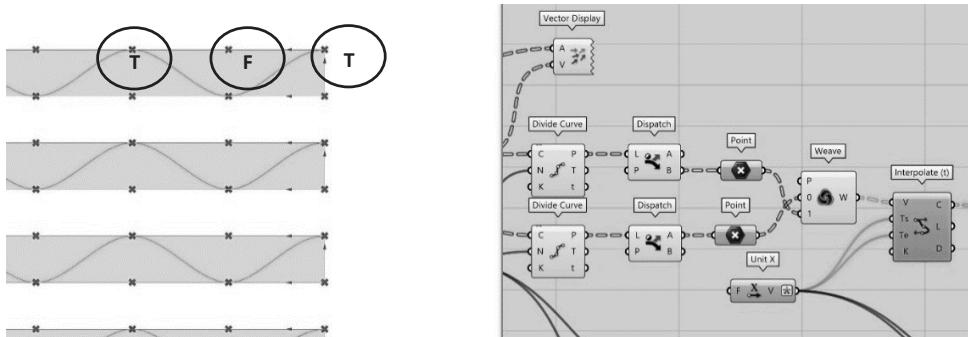
- Komponen **Dimension** digunakan untuk menghasilkan ukuran/ dimensi LCS (*Local Coordinate System*): u dan v
- Komponen **Plane Surface** digunakan untuk membuat surface dengan ukuran dimensi u dan dimensi v, nilai Plane dapat diset pada bidang x, y
- Gunakan komponen **Divide Domain²** dan **Isotrim** untuk membuat irisan/strip pada bidang dengan ketentuan strip sesuai dengan Panjang permukaan. Nilai u=1, nilai v=banyaknya strip. Nilai ini harus genap (*even number*).
- Pertanyaan: *mengapa nilai ini harus genap?*

Langkah 2: Membagi Bidang Menjadi Strip dan Membagi Setiap Strip Menjadi Titik- Titik Segmen



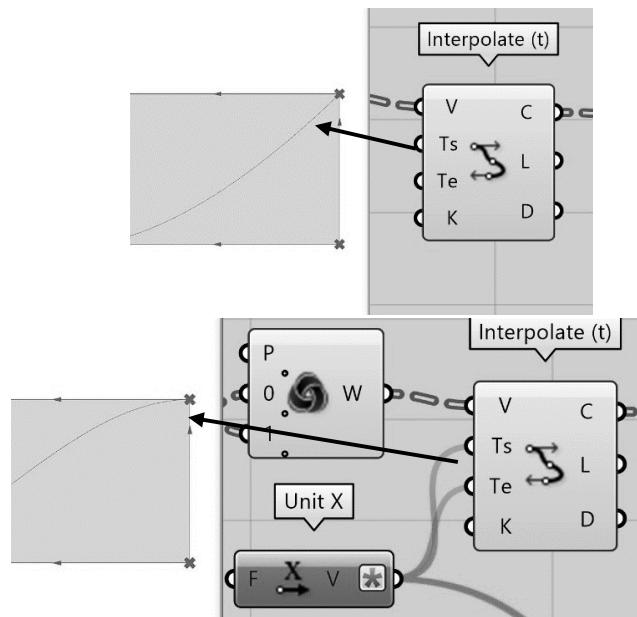
- Setelah dibagi menjadi strip menggunakan komponen **Isotrim**, kita akan bagi dua strip tersebut menggunakan komponen **Dispatch**, kita akan mengerjakan bagian A dari hasil komponen Dispatch ini. Output A berupa Surface (strip) yang akan kita bagi menjadi beberapa segmen pada arah memanjangnya (garis atas dan garis bawah).
- Gunakan **Deconstruct Brep** untuk mengekstrak Edge. Kita gunakan **List Item** untuk mengekstrak Edge/ garis yang Panjang, yakni nomor indeks 0 dan 2.
- Sebelumnya, menggunakan **Vector Display**, anda bisa melihat arah garis atas dan garis bawa berbeda, karena itu, salah satu garis ini harus dibalik arah vektorinya. Kita gunakan **Flip Curve** baik untuk nomor indeks 0 atau 2 (*mengapa Flip Curve hanya diterapkan pada salah satu indeks?*)
- Kedua indeks 0 dan 2 dibagi menjadi beberapa segmen menggunakan **Divide Curve**. Parameter pembagian segmen harus genap.

Langkah 3: Membuat Titik Segmen Selang-Seling dan Membuat Kurva Interpolasi



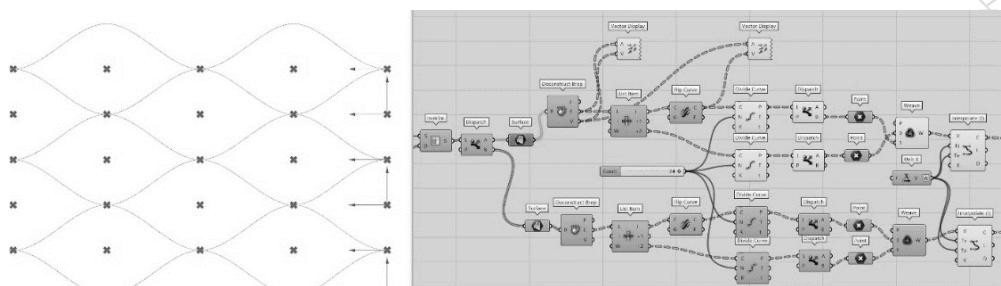
- Setelah masing-masing segmen (garis atas dan garis bawah) dibagi menjadi beberapa titik segmen (jumlah genap) dan arah vector garis atas dan garis bawah sudah sama, maka selanjutnya, pada masing-masing titik segmen harus dibuat selang-seling. Kita gunakan komponen **Dispatch** untuk menyaring data titik-titik berdasarkan T/F (True/ False). Hal yang sama diberlakukan pada garis bawah.

- Gunakan komponen Weave untuk menghubungkan titik-titik hasil Dispatch.
- Gunakan Interpolate (t)- tangensial untuk membuat kurva.
- Pada input Ts dan Te (Tangen Start dan Tangen End), isi dengan nilai yang dihasilkan oleh komponen Unit X (karena segmen mengarah pada sumbu X) dan beri expression $-x$



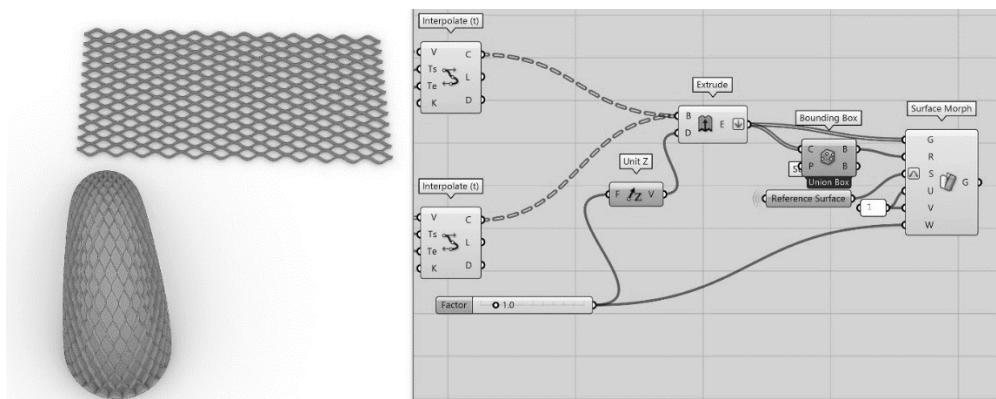
- Perbedaan tanpa nilai tangen tanpa $-x$ (kiri) dan dengan nilai $-x$ (kanan).

Langkah 4: Duplikasi Algoritma



- Duplikasi algoritma pada strip A ke strip B hingga menghasilkan seperti di gambar.

Langkah 5: Extrude Pola Gelombang dan Petakan Pada Obyek referensi

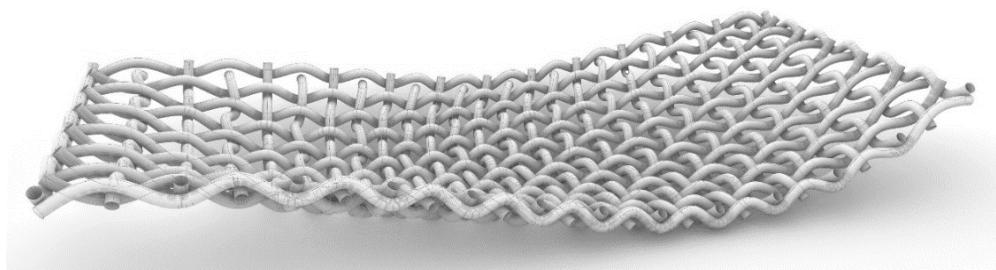


- Extrude hasil interpolasi pada sumbu Z.
- Gunakan komponen **Morph Surface** untuk memetakan hasil ekstrude pada bidang ke surface/ obyek referensi. Input G= obyek pola gelombang (Flatten), R= reference, gunakan komponen **Bounding Box** (parameter: Union Box) pada obyek hasil extrude, S= surface referensi yang sudah *di-reparameterize*. Masing-masing nilai u dan v adalah 1, karena surface referensi telah di-reparameterize, dan nilai w= tinggi ekstrude gelombang.

7.8. Anyaman (*Weaving*)

Pengembangan lain dari pola adalah pola anyaman/ *woven pattern*.

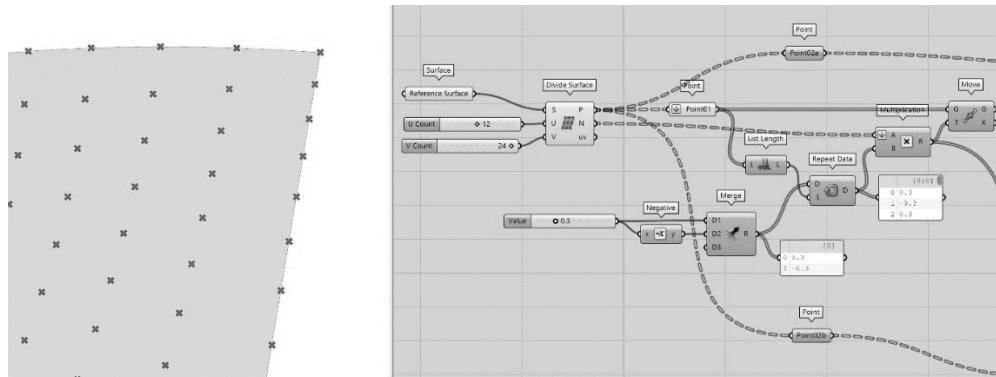
- Setiap titik pada grid diduplikasi dengan dua nilai (-t dan +t) dimana t adalah ketinggian atau ketebalan anyaman.



Gambar 178. Pola Anyaman

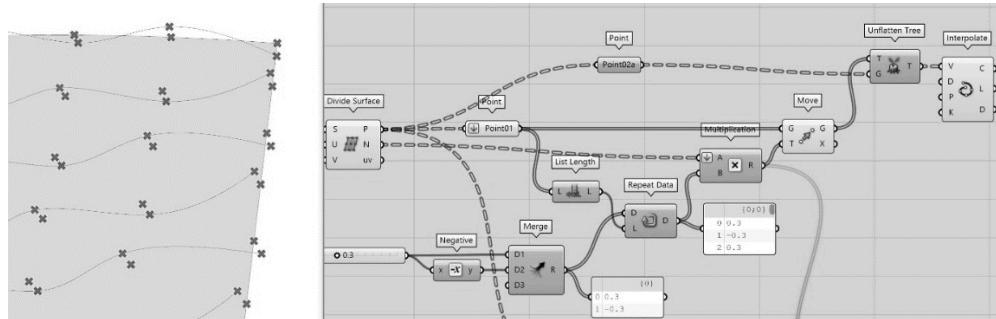
- Pada masing-masing arah, setiap titik hasil duplikasi dibuat kurva dan diberi penebalan dengan obyek pipa.

Langkah 1: Membagi Surface dan Menentukan Naik Turun Titik-Titik Grid.



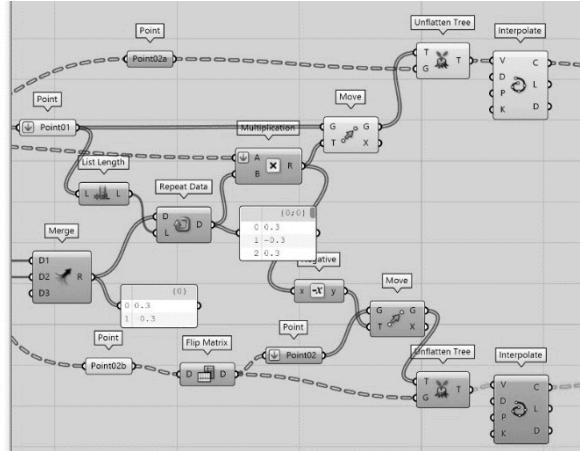
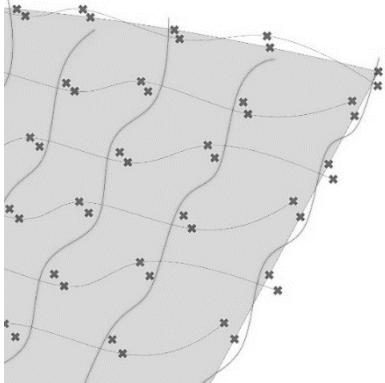
- Komponen **Divide Surface** digunakan untuk membuat titik-titik grid pada arah u dan v. Hasil dari komponen ini adalah kumpulan titik P dan titik-titik normal N.
- Tentukan banyaknya titik P (*flatten*) dengan komponen **List Length**.
- Komponen **Repeat Data** digunakan untuk menentukan nilai tertentu pada sekumpulan list. Input D adalah data yang ingin dimasukkan, input L adalah banyaknya list.
- Disini kita akan menentukan seberapa tinggi/ tebal anyaman. Pada setiap titik, harus ada titik dengan nilai + (mengarah ke normal +) dan nilai-(mengarah ke normal -) dengan nilai yang sama.
- Jadi kita masukkan pada input D dua nilai, yakni +N dan -N. Komponen Repeat Data akan merepetisi nilai ini sebanyak data atau sepanjang list.
- Kita bisa masukkan nilai 0.3 dan komponen **Negative** agar memiliki dua nilai dan masukkan ke komponen Repeat Data.
- Gunakan komponen **Multiplication** untuk mengalikan nilai hasil **Repeat Data** dengan output N hasil dari **Divide Surface** (*flatten*).
- Gunakan komponen **Move** untuk memindahkan titik-titik P dengan factor hasil multiplikasi output N pada **Divide Surface** dengan **Repeat Data**.

Langkah 2: Membuat Kurva Interpolasi Pada Arah V



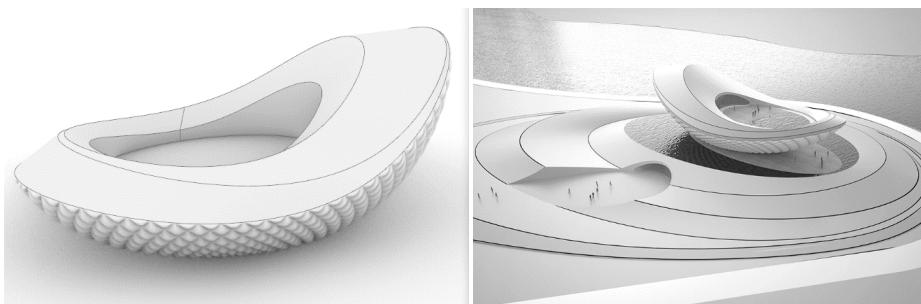
- Fokus pada Point2a seperti gambar di atas. Titik-titik ini adalah titik-titik hasil **Divide Surface**.
- Gunakan komponen **Unflatten Tree** dengan obyek T adalah titik-titik hasil pemindahan naik-turun komponen **Move** dan G-guidance adalah titik-titik hasil **Divide Surface**.
- Komponen **Interpolate** digunakan untuk menghubungkan hasil dari **Unflatten Tree**.

Langkah 3: Membuat Kurva Interpolasi Pada Arah U



- Fokus pada Point 2b seperti pada gambar di atas. Gunakan **Flip Matrix** untuk merubah orientasi pola titik-titik grid. Pindahkan titik-titik ini sesuai dengan hasil perkalian (lingkaran biru) namun dengan nilai (-). Titik-titik yang dipindahkan harus di-flatten.
- Hasil dari **Flip Matrix** dihubungkan ke komponen **Unflatten Tree** sebagai guidance dan titik-titik hasil komponen **Move** sebagai input **T**.
- Komponen **Interpolate** digunakan untuk membentuk kurva.
- Gabungkan hasil **Interpolate** dan gunakan **Pipe** untuk membuat pipa.

7.9. Smooth Panel



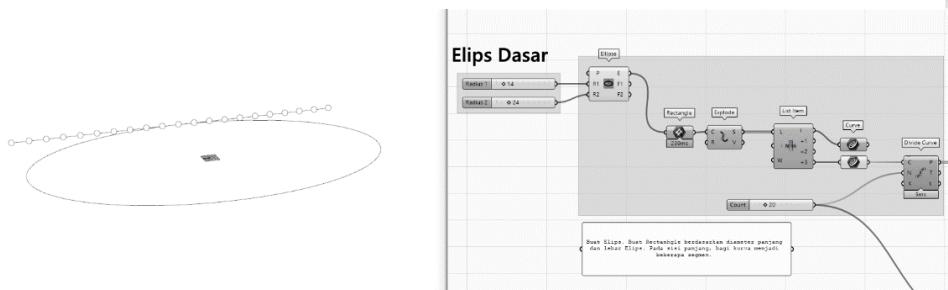
Gambar 179. Arc River Pavilion oleh Asymptote Architect

Membuat panel pada permukaan non-planar.

Prosedur atau algoritma:

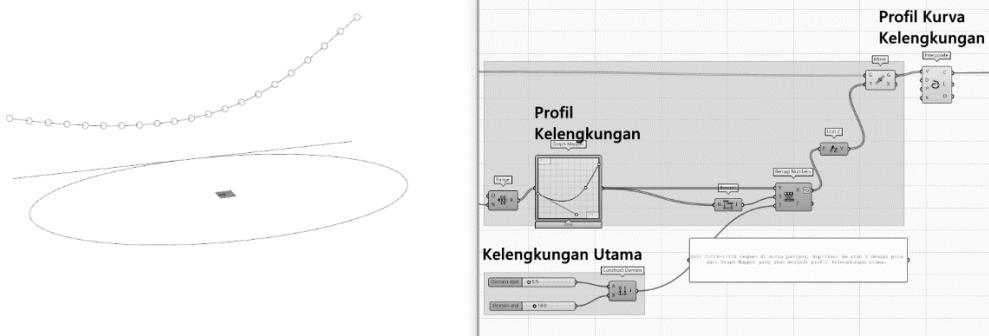
1. Membuat bentuk elips dasar pada bidang horizontal
2. Membuat profil untuk surface enclosure
3. Membuat panel atas enclosure

Langkah 1: Membuat Profil Ellips Dasar

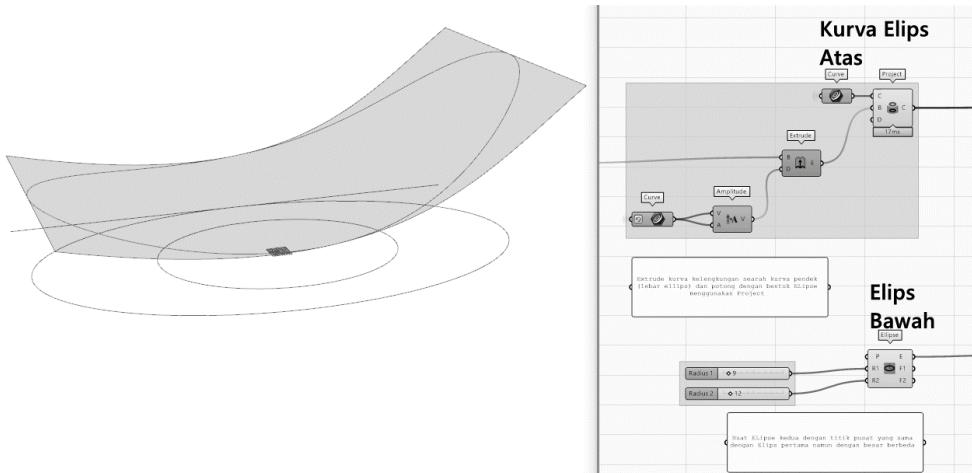


- Buat Ellipse dengan parameter R1 dan R2 (Ini akan menentukan besar atau dimensi paviliun).
- Buat kerangka Ellipse menggunakan Rectangle, ekstrak sisi panjang dari Rectangle yang juga merupakan proyeksi lengkung panjang dari Ellipse.
- Bagi Edge sisi panjang tersebut menjadi beberapa titik segmen menggunakan Divide Curve. Ini akan menjadi input untuk kelengkungan vertikal dari Paviliun.

Langkah 2: Membuat Profil Ellips dengan Kelengkungan

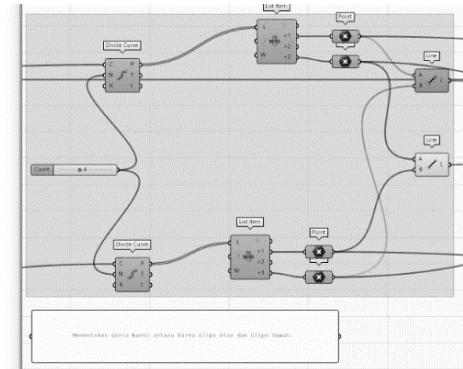
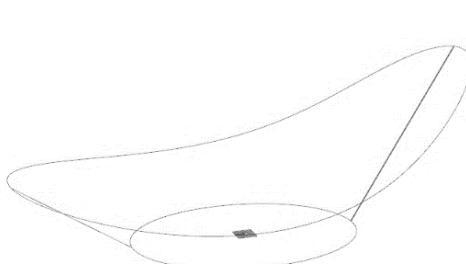


- Profil kelengkungan ditentukan menggunakan **Graph Mapper** atas input titik-titik segmen dari **Divide Curve**.
- **Remap Number** digunakan agar kita punya kontrol atas dua parameter kelengkungan vertikal pada nilai/skala sebenarnya.
- Hasil dari **Remap Number** dapat di-Reverse sesuai arah kelengkungan vertikal dan kemudian dipindahkan ke atas (arah Z).
- Buat Curve hasil dari **Remap Number** dengan **Interpolate**.

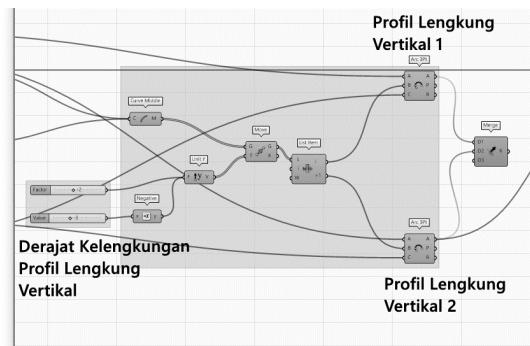
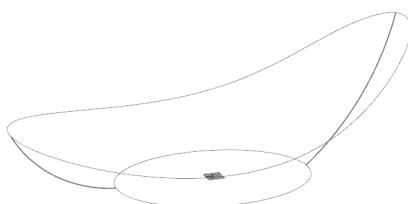


- Menggunakan **Project**, buat duplikasi Ellipse dasar yang diproyeksikan pada bidang hasil kurva yang di-extrude. Ini akan menghasilkan kurva Ellipse horizontal yang melengkung.
- Buat **Ellipse** baru yang merupakan duplikasi yang diperkecil dari Ellipse dasar.

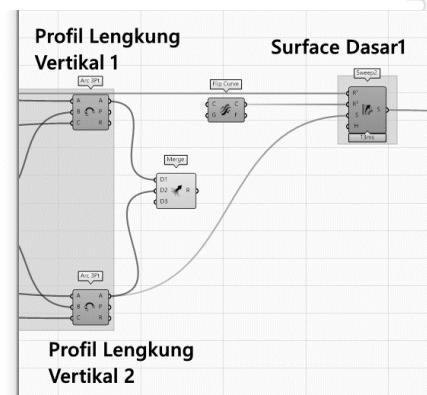
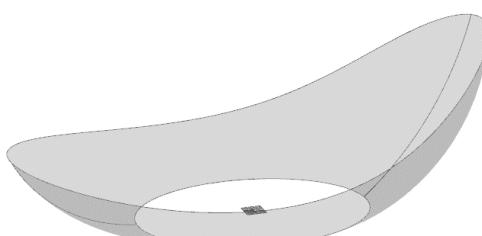
Langkah 3: Membuat Garis Bantu Antar Ellips



- Buat garis bantu yang menghubungkan Ellipse atas dan Ellipse bawah, untuk panduan membuat Arc.

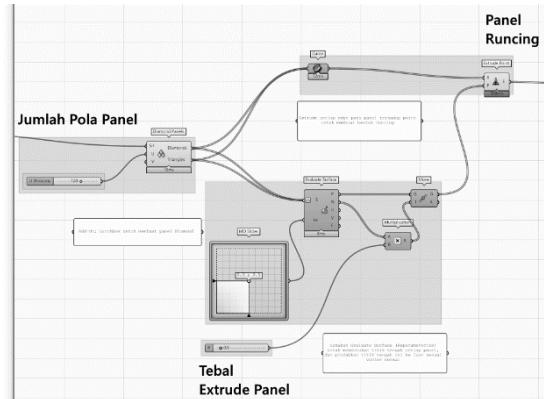
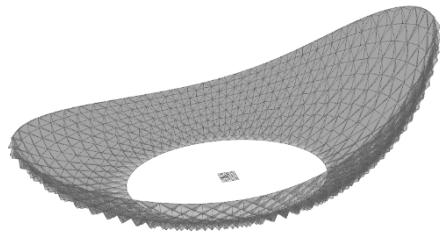


- Buat dua Arc yang masing-masing menghubungkan Ellipse atas dan Ellipse bawah.

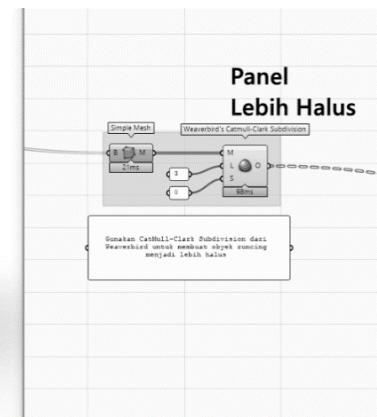
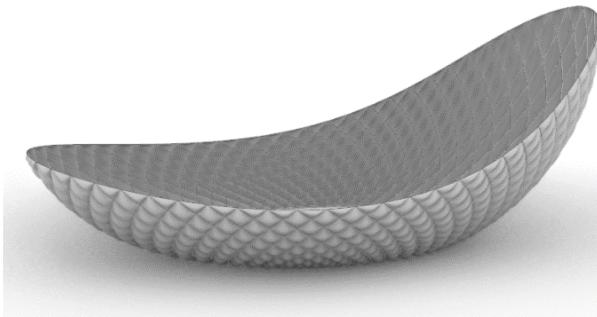


- Gunakan Sweep2 untuk membuat Surface dengan input: Ellipse atas, Ellipse bawah, dan Arc.

Langkah 4: Membuat Panel Permukaan



- Gunakan Plug-in LunchBox untuk membuat panel Diamond.
- Evaluate Surface untuk menentukan titik tengah setiap panel dan Point Extrude untuk membuat ‘duri’.
- Gunakan Plug-in Weaverbird- Cattmull-Clark Subdivision untuk membuat duri lebih halus (*smooth*).



- Ini adalah panel untuk bagian sisi bangunan.

8. TRANSFORMASI

Pada bab sebelumnya kita mempelajari aplikasi algoritma yang melibatkan operasi matematika, logika dan struktur data untuk memanipulasi data pada *Surface Geometry*. Sekarang kita akan mempelajari *Geometric Transformation* yang terdiri dari:

1. Transformasi Euklid: transformasi geometri yang mempertahankan dimensi dan sudut-sudut obyek asal: translasi, rotasi, refleksi.
2. Transformasi Affine: transformasi yang tidak mempertahankan dimensi dan sudut-sudut obyek asal: skala, proyeksi, *shear*.
3. Transformasi lainnya: *morph*.

Transformation	Maintain	Does not maintain
Euclidean	shape, size	position
Affine	parallelism	shape, size, position
Similarity	shape	size, position
Morph	topological relationships	geometrical properties



1. Original geometry;
2. Euclidean transformation: rotation;
3. Affine: scale. Scale is an affine transformation which maintains shape in addition to parallelism;
4. Similarity (scale + rotation);
5. Morph.

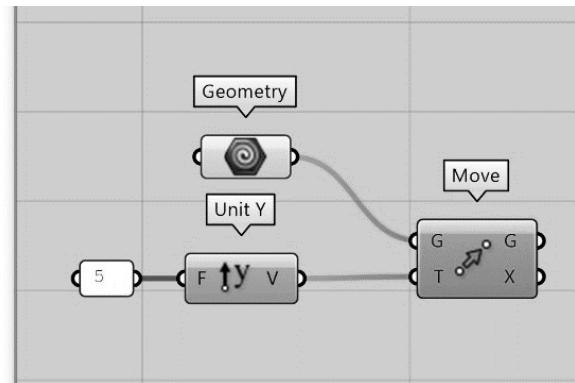
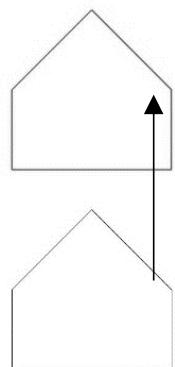
Gambar 180. Jenis-Jenis Transformasi Geometri. (Sumber: Arturo Tedesci, 2014)

Pada prinsipnya, setiap proses transformasi mengindikasikan perubahan posisi dari titik asal ke titik akhir. Perubahan ini dapat berupa penambahan atau pengurangan beberapa properti titik asal (posisi, dimensi-jika berupa obyek). Dengan demikian, ada dua faktor yang mempengaruhi proses transformasi: **arah** (*direction*), **besar** (*magnitude*).

8.1. Vector

Pada bab sebelumnya, kita mengenal beberapa komponen yang berfungsi menghasilkan vector, yang digunakan sebagai nilai untuk suatu proses transformasi.

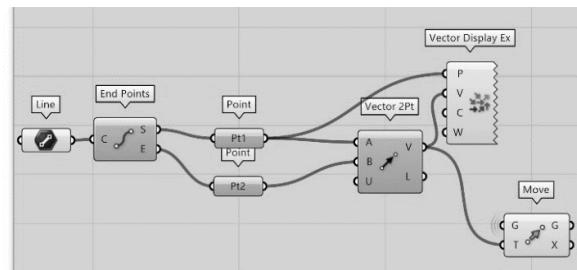
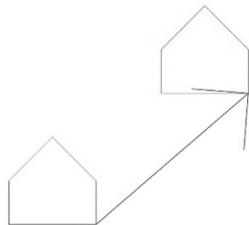
8.1.1. Unit X, Y, Z:



Gambar 181. Unit Vector X, Y atau Z

- Mendefinisikan arah vector pada tiga sumbu X, Y, Z dengan nilai F berupa skalar yang menentukan besar jarak perpindahan.

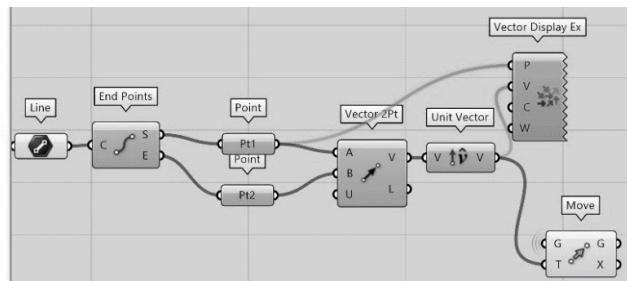
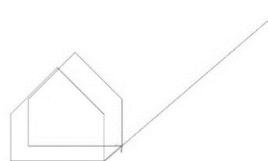
8.1.2. Vector 2Pt:



Gambar 182. Vector 2Pt

- Mendefinisikan vector berdasarkan dua titik, A= Start Point, B= End Point dan menghasilkan nilai v berupa besar dan arah.

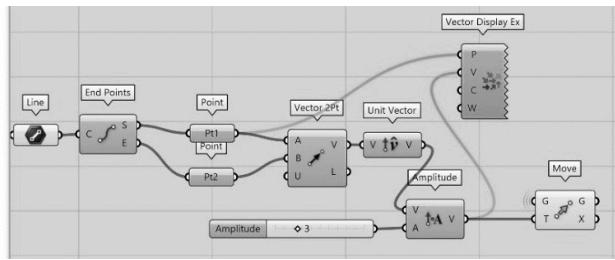
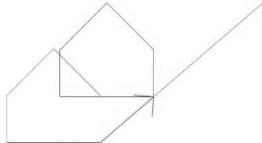
8.1.3. Unit Vector



Gambar 183. Unit Vector

- Mendefinisikan unit suatu vector (nilai=1) tanpa mengubah arah vector.

8.1.4. Amplitude



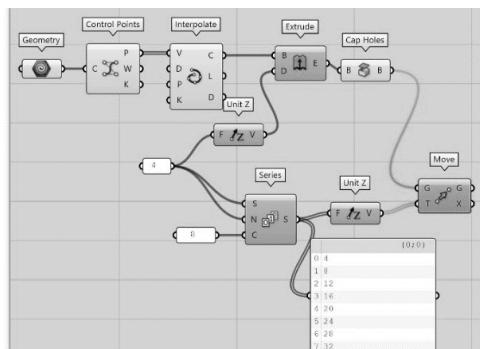
Gambar 184. Amplitude

- Memberikan nilai scalar pada suatu vector tanpa mengubah arah vector.

Pertanyaan: apa yang terjadi jika nilai pada Amplitude adalah negatif?

8.2. Transformasi Euklid

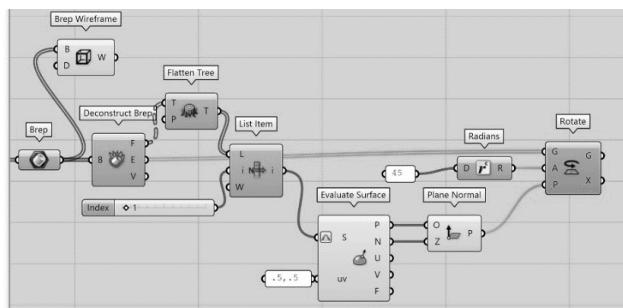
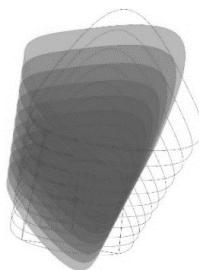
8.2.1. Translasi-Move



Gambar 185. Move

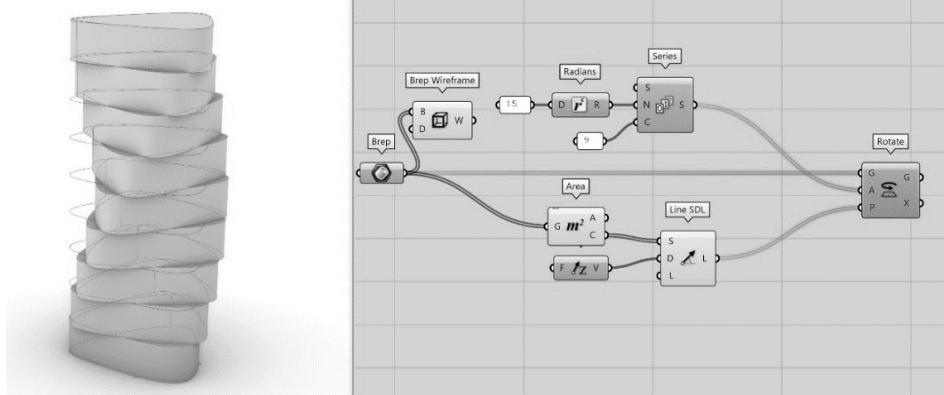
- Komponen Series digunakan untuk menentukan nilai-nilai vector untuk duplikasi.

8.2.2. Translasi- Rotate



Gambar 186. Rotate

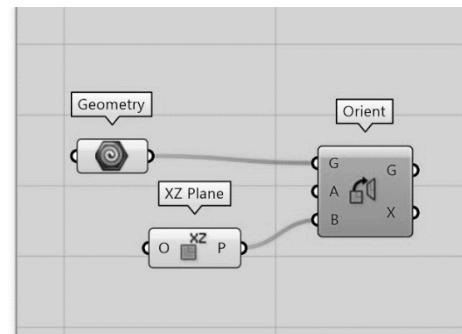
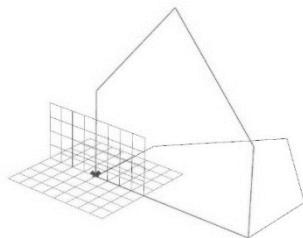
- Komponen Rotate memerlukan Geometry, Angle (sudut dalam Radian) dan Plane sebagai sumbu dan bidang putar



Gambar 187. Incremental Rotate Dengan Series

- Gunakan komponen Series untuk menentukan besar sudut putar secara incremental.
- Gunakan komponen Area untuk menentukan titik pusat setiap obyek hasil duplikasi (dalam contoh di atas adalah setiap massa lantai), titik ini digunakan sebagai titik sumbu putar dengan sumbu putar Z pada komponen Line SDL yang digunakan sebagai input Plane ada komponen Rotate.

8.2.3. Translasi–Orient



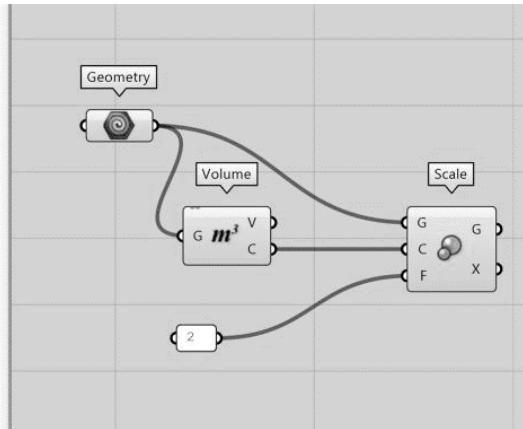
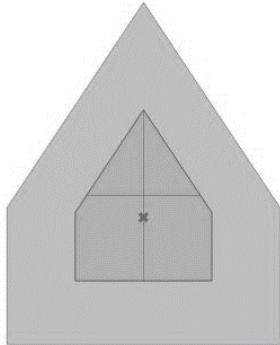
Gambar 188. Orient

- Menggabungkan translasi dan rotasi dalam satu komponen: **Orient**. Komponen ini memerlukan geometri dan plane pada geometri tersebut (*initial Plane*) dan orientasi dan bidang target (*Target Plane*).

8.3. Transformasi Affine

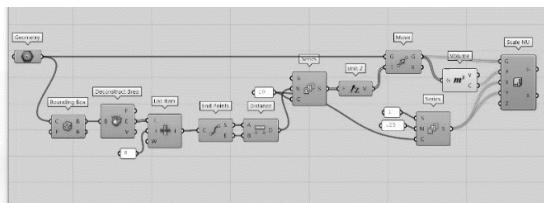
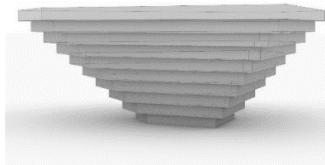
Yang termasuk transformasi Affine adalah: **Scale**, **Shear**, **Projection** dan **Mapping**.

8.3.1. Scale dan Scale NU (Non Uniform)

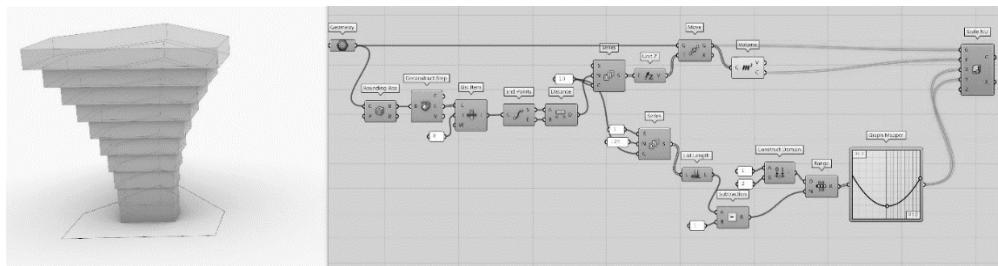


Gambar 189. Scale

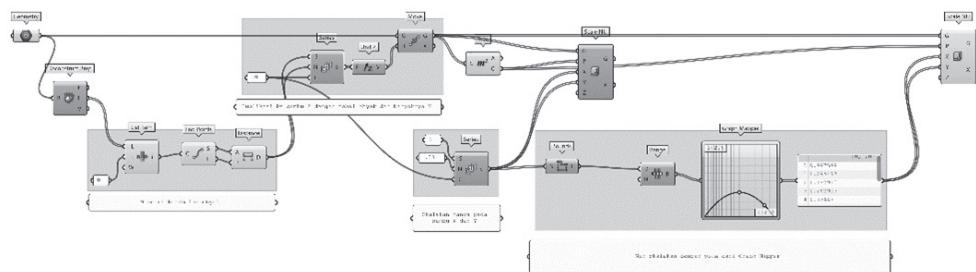
- Komponen **Scale** berfungsi men-skalakan obyek. Perhatikan input **C (Center)** adalah pusat skala, **F (Factor)** adalah nilai skala: <1 memperkecil; >1 memperbesar.
- Nilai skala ini bisa juga berupa nilai jamak/ tidak tunggal, seperti dihasilkan oleh komponen **Series**, **Graph Mapper**, **Range** dan lainnya.



- **Scale NU** digunakan kalau kita ingin melakukan skala hanya pada sumbu tertentu (*Non Uniform*). Misalnya kita ingin membuat duplikasi obyek dengan skala pada X dan Y namun ketebalan obyek (nilai pada sumbu Z) tetap 1. Kita bisa gunakan komponen **Scale NU**.
- Duplikasikan obyek pada sumbu Z menggunakan komponen **Move**. Nilai faktor perpindahan digenerasi oleh komponen **Series**, dimana nilai awal **S (Start)** adalah 0, nilai **N (Step)** adalah total tinggi obyek asal, dan nilai **C** adalah banyaknya obyek duplikasi.
- Setelah obyek diduplikasi selanjutnya, setiap hasil duplikasi diskalakan dengan komponen **Scale NU** titik pusat skala ada di titik pusat obyek masing-masing. Komponen **Series** digunakan untuk memproduksi nilai skala pada sumbu X dan Y pada komponen **Scale NU**.
- Komponen **Bounding Box** digunakan untuk menentukan ketinggian obyek asal. **Deconstruct Brep** komponen ini dan cari jarak atau ketebalan pada *vertical edge* menggunakan komponen **List Item**, **Endpoints** dan **Distance**.



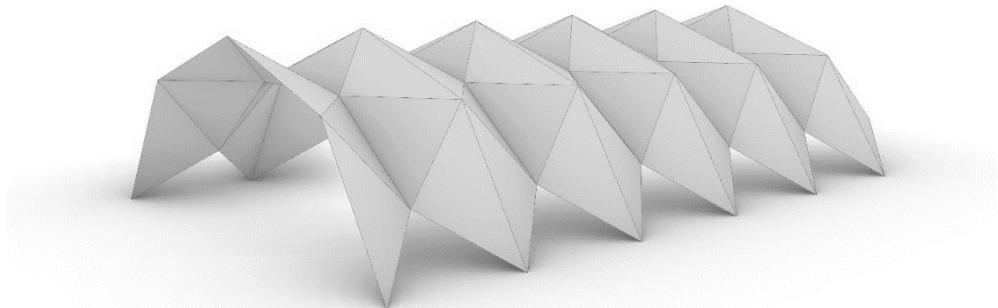
- Pengembangan selanjutnya adalah bila nilai skala ditentukan oleh kurva dari komponen Graph Mapper dan tidak linier seperti di contoh pertama.



- Implementasi Move, Rotate, Scale pada generasi beberapa bentuk seperti terlihat di atas.

8.4. Bentuk Lipat (*Folded Structure*)

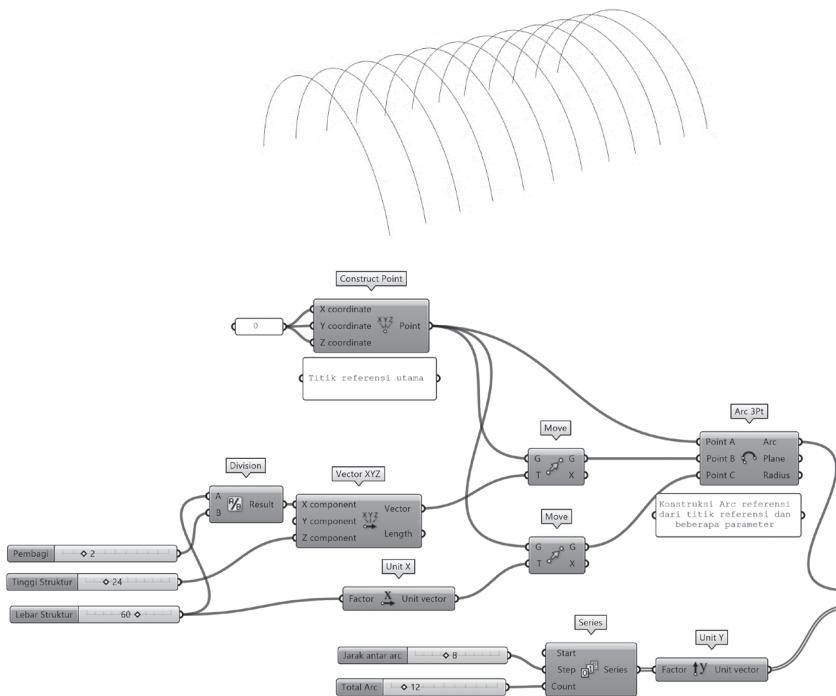
R AR-4214 THU



Gambar 190. Pola Lipat (*Folded*)

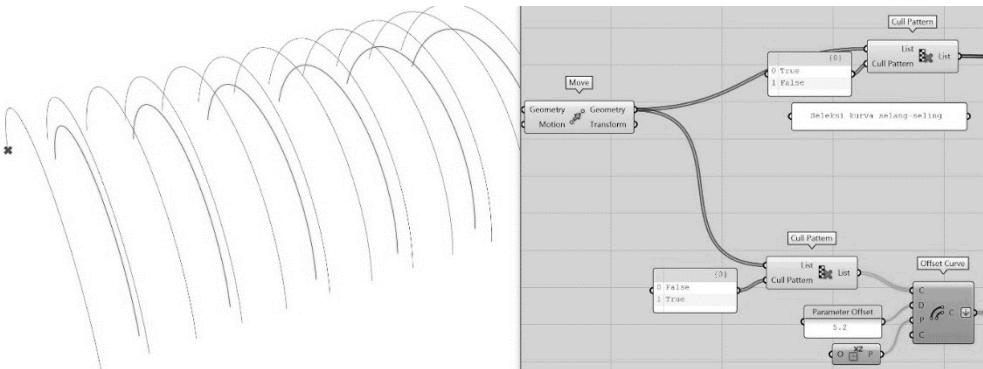
- Membuat konstruksi lipat terowongan (*folded tunnel*) dengan menggunakan satu titik referensi dan semua dilakukan di GH. Obyek dasar berupa terowongan setengah lingkaran menggunakan Arc.

Langkah 1: Mendefinisikan Titik Referensi, Arc dan Struktur Terowongan



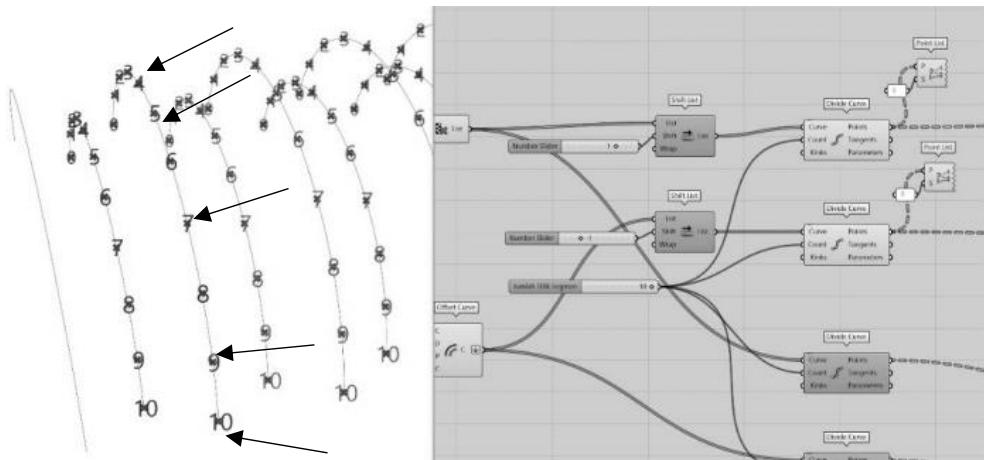
- Tentukan titik referensi dengan **Construct Point**. Ini akan menjadi Point A pada bentukan Arc menggunakan komponen **Arc 3pt**. Point B didapat dengan cara menduplikasi Point A yang dipindahkan (**Move**) ke arah sumbu Z dengan parameter Tinggi Struktur. Point C didapat dengan cara menduplikasi Point A yang dipindahkan pada arah sumbu X dengan parameter Lebar Struktur.
- Selanjutnya, Arc01 diduplikasi ke arah sumbu Y sejumlah 12 buah dengan jarak antar Arc=8 meter menggunakan komponen **Series**, **Unit Y** dan **Move**.

Langkah 2: Membuat Duplikasi Arc



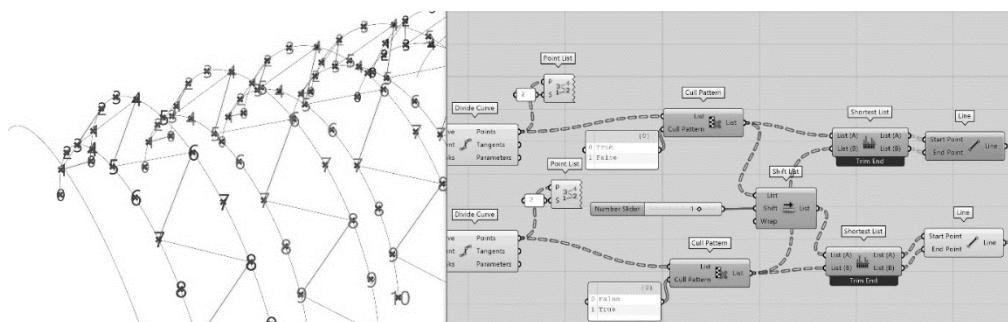
- Arc hasil duplikasi harus diseleksi selang-seling dan dibagi dua: kumpulan arc 1 menggunakan Cull Pattern dengan pola (True, False), kumpulan arc2 menggunakan Cull Pattern dengan pola (False, True). Kumpulan Arc2 ini kemudian di-offset ke dalam (pada arah XZ) dengan parameter ketebalan lipat atau parameter offset= 5.2 (anda dapat tentukan sendiri kedalaman lipat ini).

Langkah 3a: Menyusun Titik-Titik Segmen



- Fokus pada Sebagian bidang folded. Gunakan komponen Shift List untuk menetapkan Arc-Arc yang akan dihubungkan. Nilai Shift List arc besar=+1, nilai Shift List arc kecil=-1. Masing-masing arc kemudian dibagi menjadi beberapa segmen dengan Divide Curve. Nilai banyaknya segmen menjadi parameter banyaknya sudut lipat antar arc. Perhatikan bahwa urutan dan arah indeks pada setiap arc baik yang besar maupun yang kecil sudah berurutan dan searah.

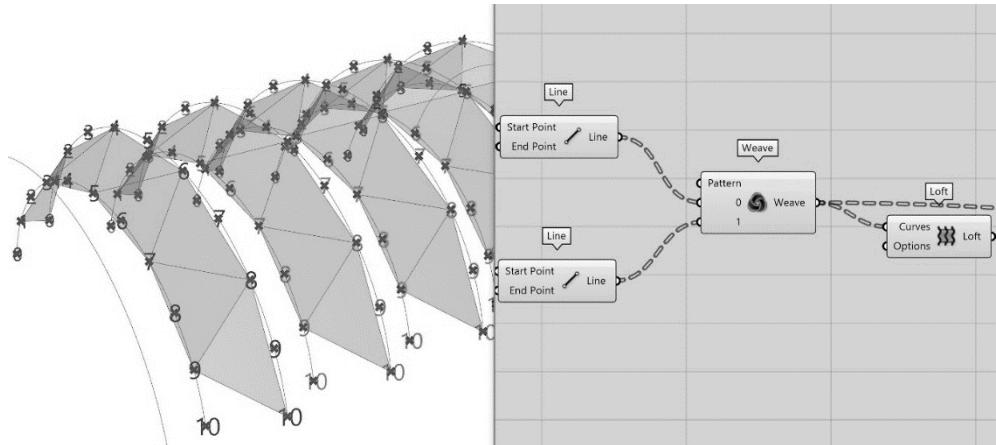
Langkah 4a: Menyusun Titik-Titik Segmen dan Membuat Garis Penghubung Antar Titik



- Lanjutan dari Langkah 3a, masing-masing titik pada segmen arc diseleksi dengan Cull Pattern dengan perbedaan nilai T/F. Pertanyaan: mengapa nilai pola T/F berbeda antara segmen arc besar dan segmen arc kecil?

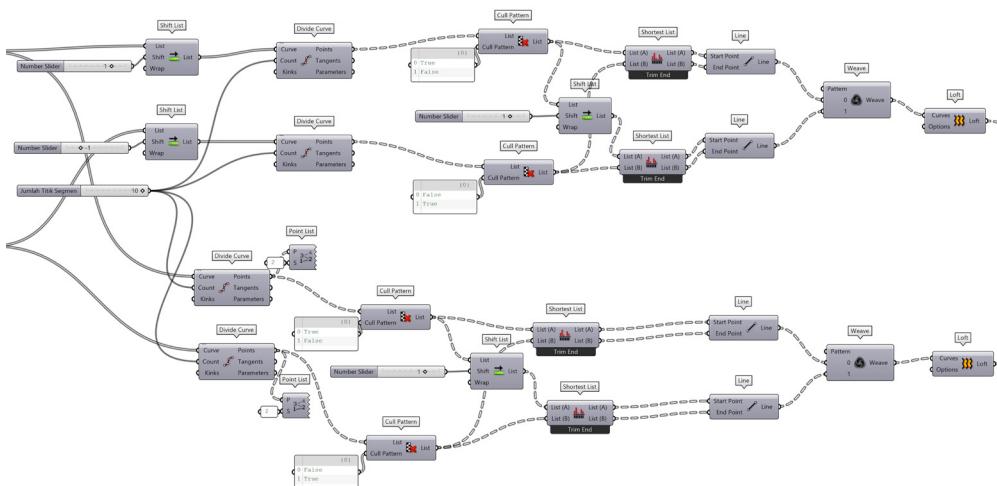
- Pada masing-masing segmen, gunakan komponen Shortest List (Trim End) untuk mencocokkan data list titik antara titik-titik segmen arc besar dan segmen arc kecil.
- Gunakan Line untuk menghubungkan antar titik.

Langkah 5a: Membuat Jalinan Garis dan Permukaan Bidang



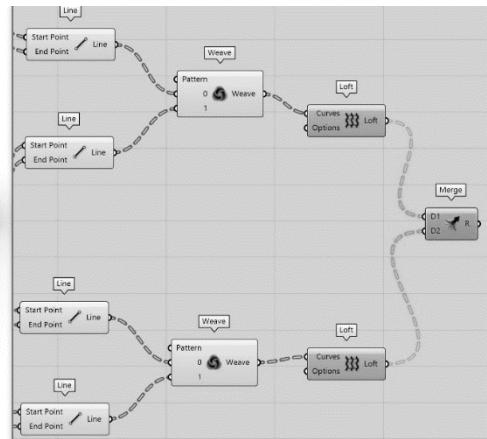
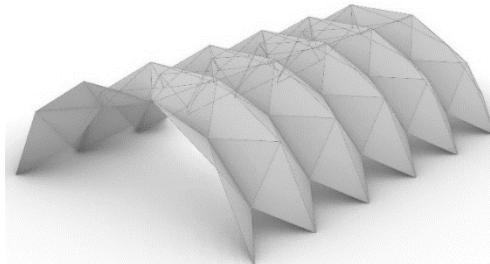
- Gunakan Weave pada masing-masing garis hasil Langkah 4a dan komponen Loft untuk membuat permukaan.

Langkah 5b: Menentukan dan Menyusun Titik-Titik Segmen dan Membuat Garis Antar Titik Segmen



- Hampir sama dengan Langkah 3a dan duplikasi Langkah 4a dengan sedikit modifikasi pada susunan list.

Langkah 6: Gabungkan Semua



- Gabungkan hasil Weave dan Loft.