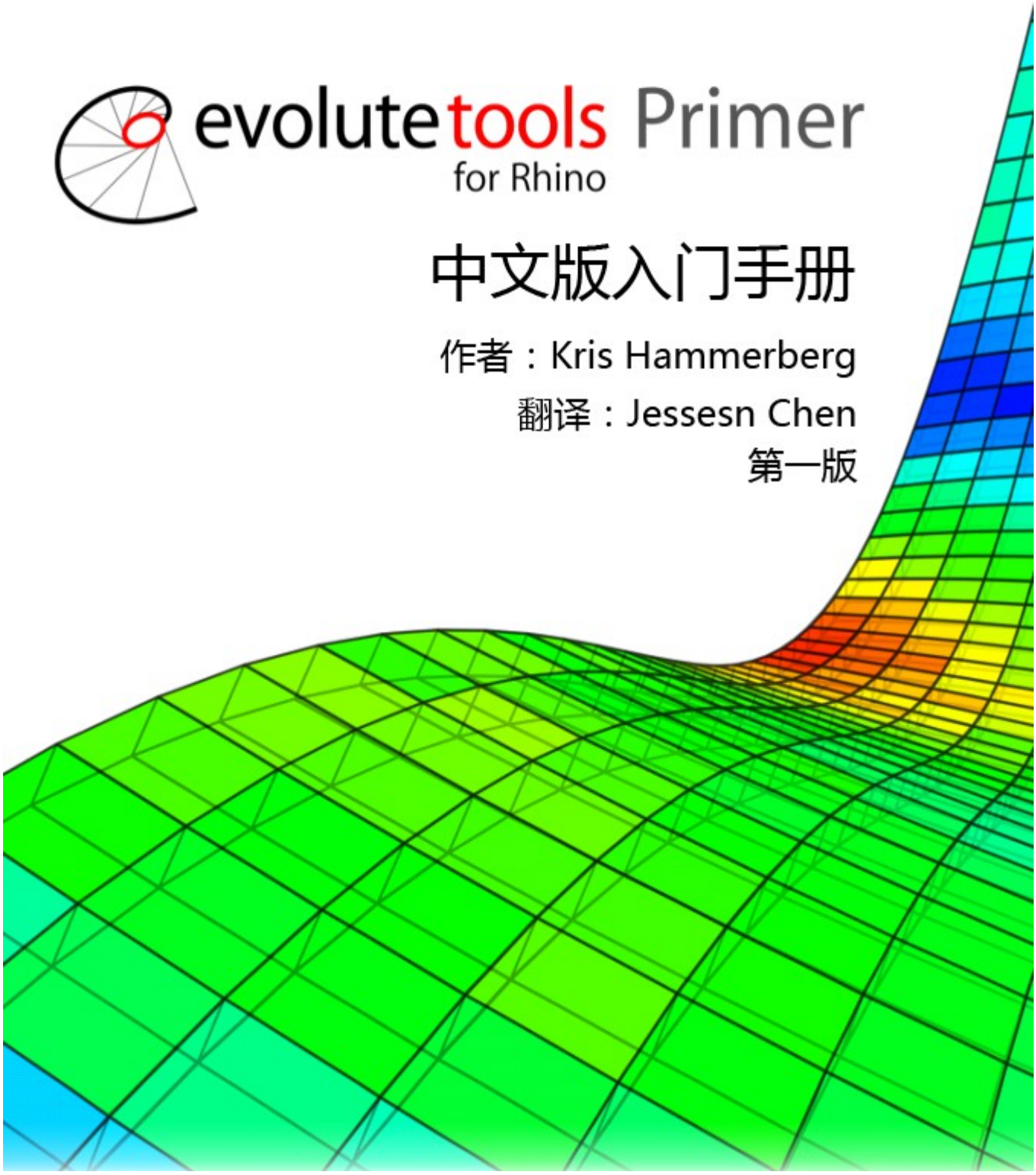
evolutetools Primer
for Rhino

中文版入门手册

作者：Kris Hammerberg

翻译：Jessesn Chen

第一版



目录

1. 介绍: EvoluteTools 入门手册

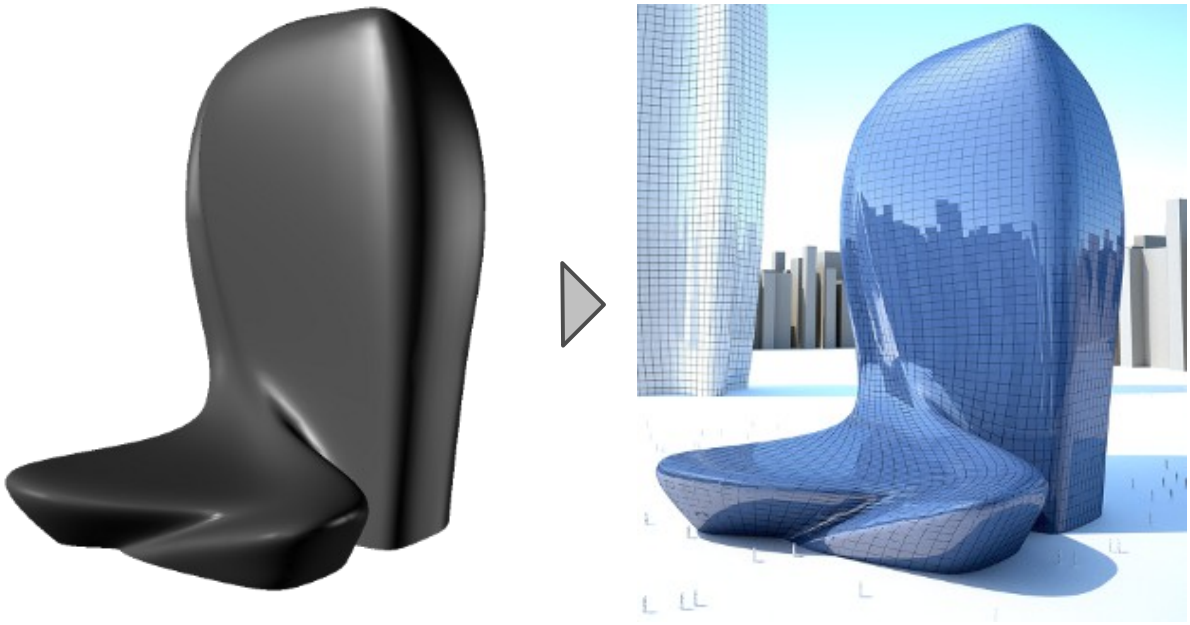
你手中（或是在屏幕上显示）的是一本 EvoluteTools 入门手册， 主要是为了方便刚刚开始使 EvluteTools 的用户入门，也可以给熟练的用户作为参考资料，下面的章节我们会以图文并茂的方式来诠释 EvoluteTools 中的常用特征，另外还有一些提示、诀窍与技巧以帮助你能更好的掌握 EvoluteTools。无论你是否是第一次接触 EvoluteTools 我们都希望你能喜欢这部入门手册。

在我们详细的诠释这些特征之前，我想给你一些有关如何学习这边手册的一些技巧。我们希望这本手册既可以作为你的一页页学习 EvoluteTools 的入门手册也可以作为你最全面的参考书籍，显然你很幸运，编写这边手册花费了居多的努力，EvoluteTools 揽括了超多的特征！因此，随意的翻阅某些章节你会觉得在查阅一部功能清单而缺少系统的说明与解释，对于手册阅读本手册的读者来说保存激情尤为重要，我有使用 **«»** 符号标记可选读章节与子章节，当你第一次阅读时候可以越过这些章节，这些可能是一些不常用的特征或需要一定的技巧当到你需要或熟练后可以再次查阅。

2. 目的

没有人了解将一个概念转化为真实材料创建的实物中包含多少的挑战、协调与牺牲，类似将建筑设计与结构设计真正进行施工时所面对的相同问题。写这本书并不是想说明有单一、无缝、持久的材质可塑造任何的形状且可以满足所有建筑外覆盖件的要求，建筑都是分层（瓦片、面板、板瓦、防潮隔离层等）来构造的，对于平面的曲面，将其分离成小的组件来加工是一件非常容易的事情。

然而，一旦设计师设计了弯曲的造型，这个问题会变得复杂的多。最常见的解决方法是选择一些嵌板覆盖至结构支撑点上近视的逼近造型，说起来比较简单，但如何去分割一个完整的造型？如何转换一个类似下图中的自由曲面造型设计为一个真正的建筑？

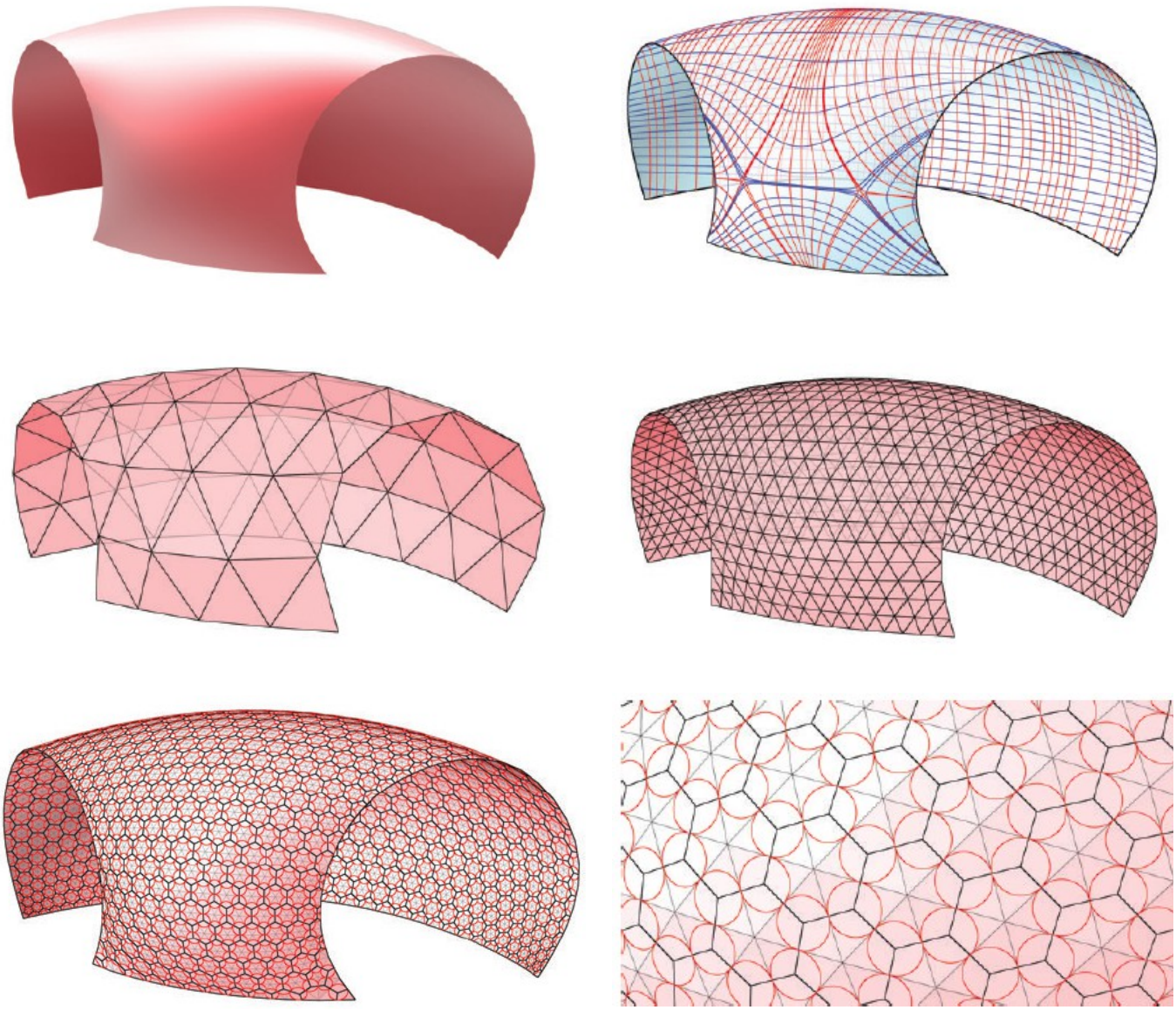


图片 *1.* – 对于这样的复杂几何造型的面板平直化工作，*EvoluteTools* 数分钟内可以完成而不是数天。

EvoluteTools 是 Rhino 平台下的一个插件，允许你在任何自由造型上创建符合你项目要求的平直构造，且你可以在数分钟内完成。是个非常强大的工具，可以将精美复杂的造型设计转换为为精美复杂的可建项目。需要得到一个矩形的嵌板吗？没有问题。 需要他们足够的平直吗？很容易，想看到平直六角形的构件吗？解决这些问题只要稍稍点击 EvoluteTools 即可完成。

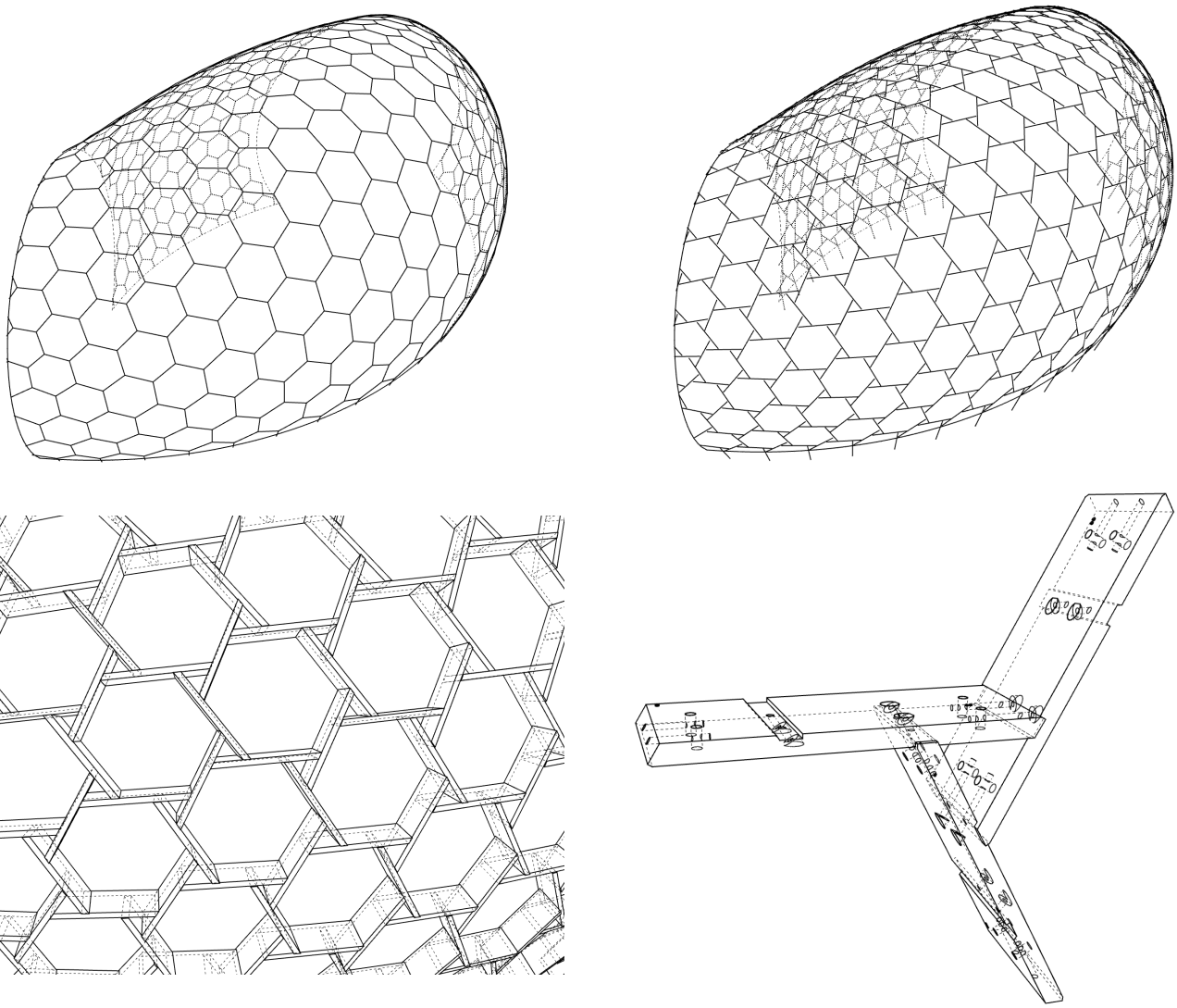
无论你是建筑师、承包商、设计师、工程师或是制造厂商，EvoluteTools 都会帮你**降低成本，缩短工期、降低风险且完美实施**。

毫无疑问 EvoluteTools 的潜能远超过曲面几何平直化，使用 EvoluteTools 网格优化，结合自身的网格偏移指令与 RhinoScript 接口，能帮助你深度的合理化多重建筑构件与建筑材料， 且能更加深入的优化。下面这些图片是由 Evolute 的合作伙伴 Pavilion Architecture 所设计的 KREOD 项目，我们作为开发团队的一部分，Evolute 优化 KREOD 所设计的曲面造型，且在其曲面上创建平直嵌板，且复杂提供这些木质构件的详细加工与装配参数。我们使用 EvoluteTools 对原始曲面进行分析、优化与平直化，这些技巧你会在这本书中学习到，最后得到这样漂亮的规则六边形网格。



图片 *2.* - 左上图：设计师所提供的原始设计曲面。 **右上图：**主曲率线分析。**中左图：**拓扑面板布局图。**左下图：** *Circle-Packing* 优化且得到最终六边形面板布局。**右下图：** 最终的 *Circle-packing* 优化面板布局，展示三角形与六边形的构成关系。

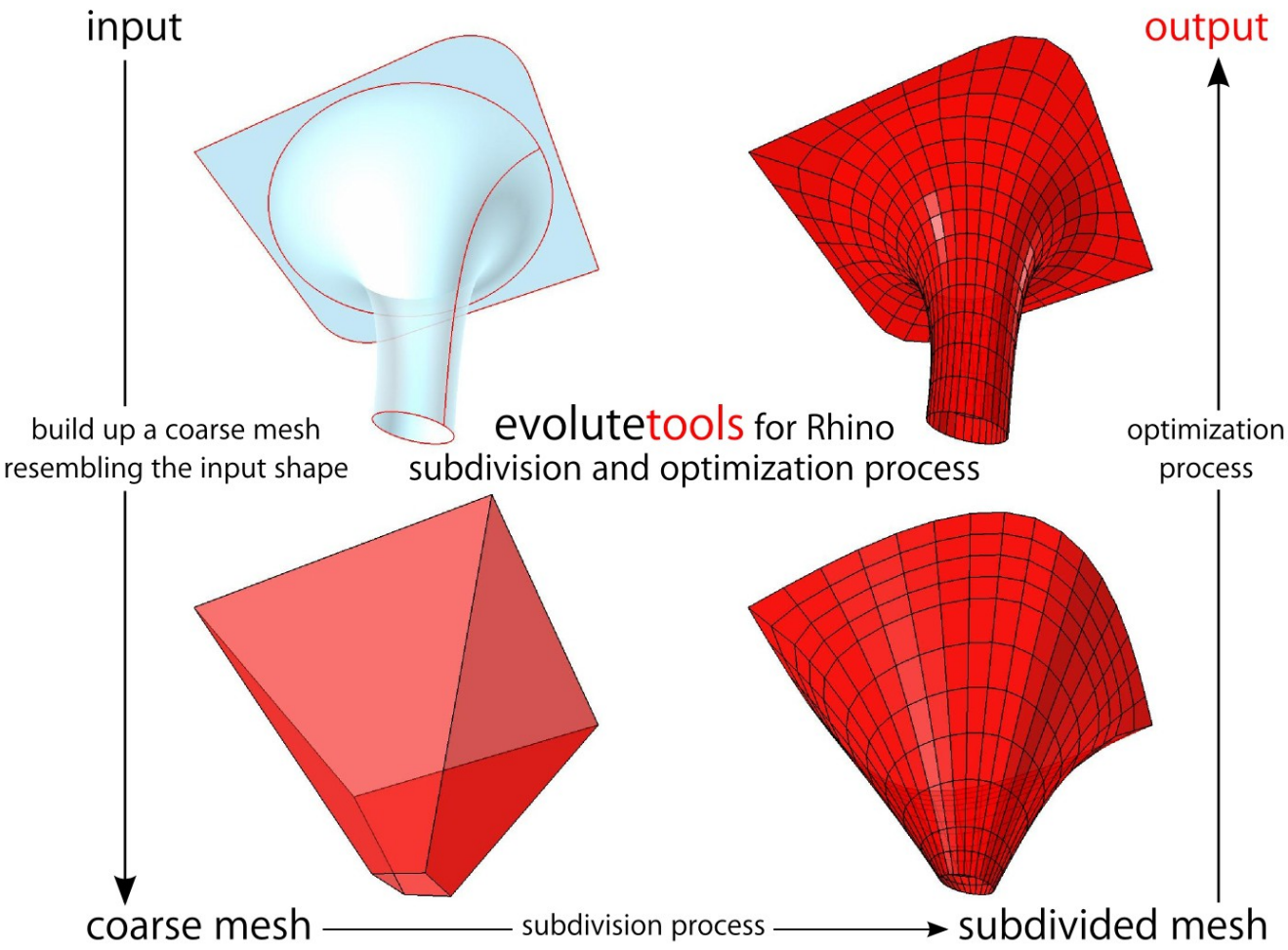
这是个非常好的网格，但要离实际建筑相差甚远，与项目工程团队 Ramboll Engineering 协商，设计一个反向的装置来连接这些结构件。[请看下面的图片]基于这个装置的设计，Evolute 以代码的方式来创建这个能够加工的漂亮图形，使用 EvoluteTools 以及其提供的 RhinoScript 接口，生产近 1000 多个组件产生，且都一一标记，然后输出这次参数**直接丢给 CNC** 进行加工。



图片 *3.*- 左上图： 六边形网格。**右上图：**旋转六边形边缘。**中左图：** 最终钢结构装配图。**中右图：**连接装置。**最下面：**在伦敦演示 *KREOD Pavilion* 现场图片，由 *Kin Ho* 提供。

3. 典型作业流程

EvoluteTools 插件是一个适合用于专业解决真正构造问题的实用工具，因此这边手册会围绕你使用 EvoluteTools 来创建面板布局时的流程*来编写。我会通过这个流程的介绍让你快速的了解整个作业流程，在后面的章节将会详细的讲解相关技巧，你同样也可以作出漂亮范例。



图片.*4 - EvoluteTools* 常用流程。

* 这个流程并非是一成不变的，当你数量掌握 EvoluteTools 之后你会发现你可以透过多种方式来完成你的项目。

3.1 第一步：定义参考对象

对于建筑师与设计师来说这是比较容易的一步，但也是十分关键的一步，这一步你需要让 EvoluteTools 知道你所需要的形体是怎样的，这样才能保证最终的结果不会偏离形体。参考对象可以是由任何的 CAD 软件程序创建，当然是可以导入进 Rhino。参考对象可以是一个网格、曲面、多重曲面、T-Splines 曲面或是任何形式的曲线与多重直线，没有任何限制。

3.2 第二步：创建“粗糙网格”

EvoluteTools 透过调控与优化网格的方式来作业，部分用户可能已经对网格非常熟练且已经了解为何把网格作为面板布局的首选对象。如果你暂时还不太明白，也没有关系，我们会在后面的章节详细讲叙，且会让你知道为何在这个环节 EvoluteTools 使用网格 5 – 网格基础知识。最开始我们只需要非常简单面数的网格且最终会得到足够完整的面板布局所需要的网格，我们把这个简单的第一个网格叫做“粗糙网格”。

粗糙网格的变化对最终的结果有很大的影响，我们会在后面的章节详细讲解 6 - 粗糙网格。但要记住最重要的一点，粗糙网格一定要简单且能与参考对象的体型接近。

3.3 第三步：细分

粗糙网格创建的尽可能的简单且尽可能是较低的面数，这些大块面都会被分成一些小的块面，这些小的块面的尺寸大概的接近最终面板布局的大概尺寸。无需手动操作，EvoluteTools 会提供一组自动化划分工具，根据你所设置的不同预置规则的参数自动划分成小的网格，这个自动的划分过程我们称之为“细分”，在 7 - 细分 章节中我们会介绍不同的细分规则。细分的步数并无限制，直到细分网格达到你的要求为止。

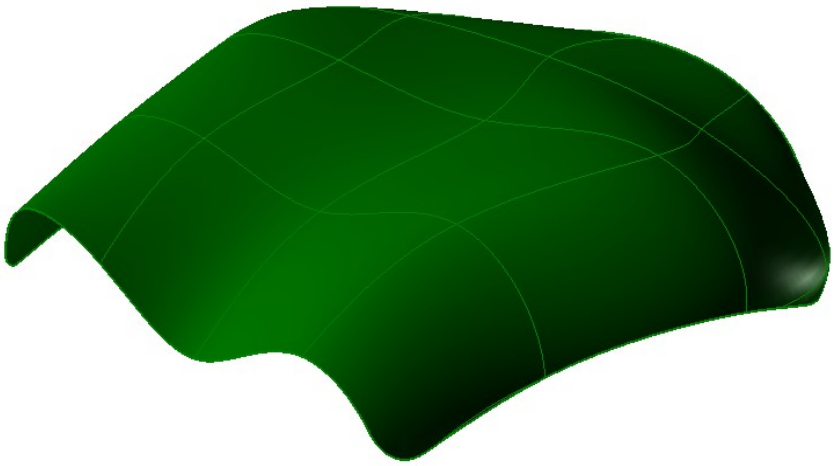
另外自动细分工具会影响到整个网格，你也可以对局部网格进行细分，允许用户使用所需要的好的结果来控制细分，我们会在章节 7 中讨论。

3.4 第四步：优化

在这一步之前，你大概还看不到你想要的结果，甚至会和原始参考物件的形状都有很大的偏差，这是 EvoluteTool 最核心的部分，次世代的优化算法会帮你调整网格的形状与原始参考对象逼近，最终完全与其吻合，你可以设置与调整不同的约束条件（例如面板的平直度）来调整面板布局以达到你项目的需求。[详情请查看 8 - 优化篇]

4. 现在 开始 实例环节 – 第一个范例

如果这是一本学习编程的书籍，那么第一步你将要学习如何在屏幕上显示出“Hello World！”，我想你会同意这是一个非常酷的范例，在入门手册文件夹[下载地址: evolute.at], 你会找到一个叫 Primer_Example1. 的 Rhino 文件，在这个文件中你能找到如下图所示的奇怪物体，如果你看到有弹出一个对话框询问你是否导入保存的优化设置（If you want to load saved optimization settings），请选择 “Yes”。

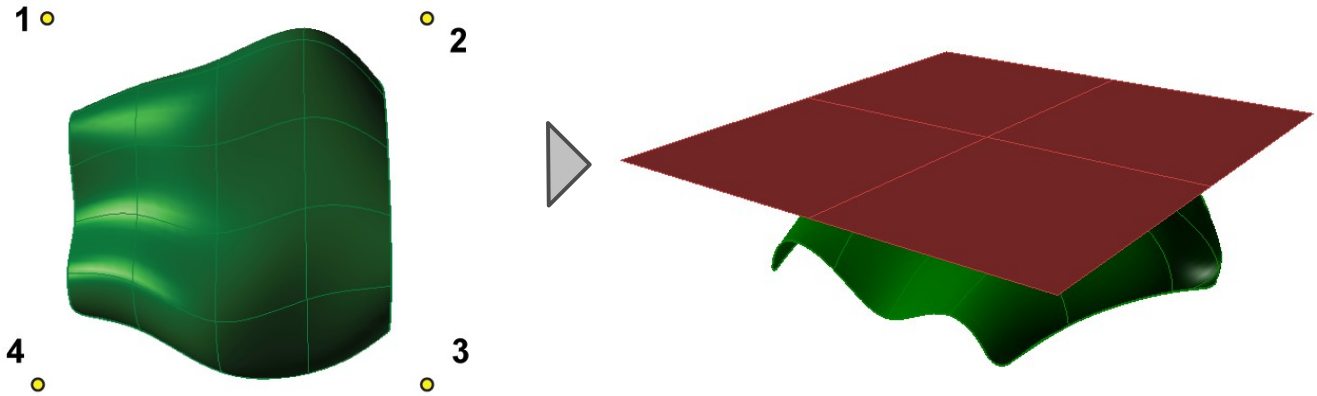


4.1 范例 1

使用 EvoluteTools 简单的 7 步就可以创建你所需要的平直布置图。

1. 根据基本的作业流程，我们第一步是定义参考对象，选择到文件中的曲面对象然后在 EvoluteTools 工具列中点击按钮：  ，或在指令行中输入 etSetReference 指令，这样就完成了参考对象的设置。根据需要，你也可以右键点击相同的图标或是在指令行输入 etClearReference 清除已经存在的参考对象。

2. 接着创建一个粗糙网格，创建粗糙网格的原则是尽可能的精简。切换至 Top 视图，点击  或在指令行执行 3DFace 指令，由现有的 4 点创建一个方形的四边网格.[如下图所示]

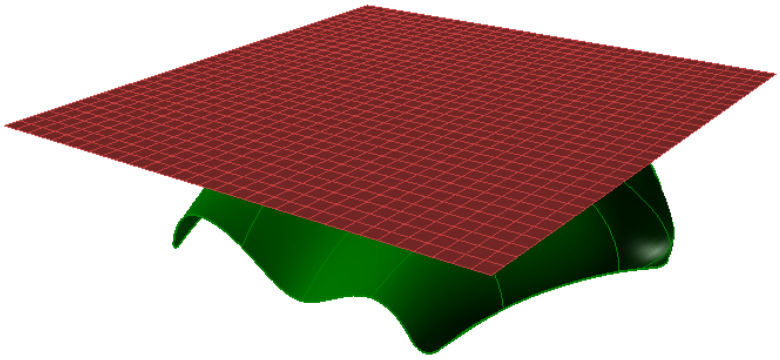


现在你得到了一个尺寸接近 50mX50m 的简单网格，这么大的形体对于制造、施工都是一个很大的挑战，且和我们想要的最终结果完全不同，我们需要更小的面板！

3. 现在你要使用 Catmull-Clark（四边网格通用细分规则）方式细分网格，选择粗糙网格，然后点击  或执行 etSubDivide 指令，指令行将会询问你选择哪一种细分规则，默认的规则为 Catmull-Clark， 如果你要使用其他的细分规则，请点击所显示的规则名字或是敲入 R，这样会列出所有可用的细分规则。

   会出现一些选项，在这个范例我们仅使用 Catmull-Clark, 选择 Catmull-Clark 然后回车。你会看到在原来 1 个网格面相同的位置新生产 4 个网格，[如图所示]

4. 对于实际建筑来说使用这些面板还是太大，你需要重复执行细分，继续选择原始的粗糙网格，然后继续 4 次细分。
   [提示: 按[回车]会重复执行前一次的指令，立即再次支持细分指令]。
   最后你会得到 1024 块尺寸大概为 1.6mx1.6m 的网格面。



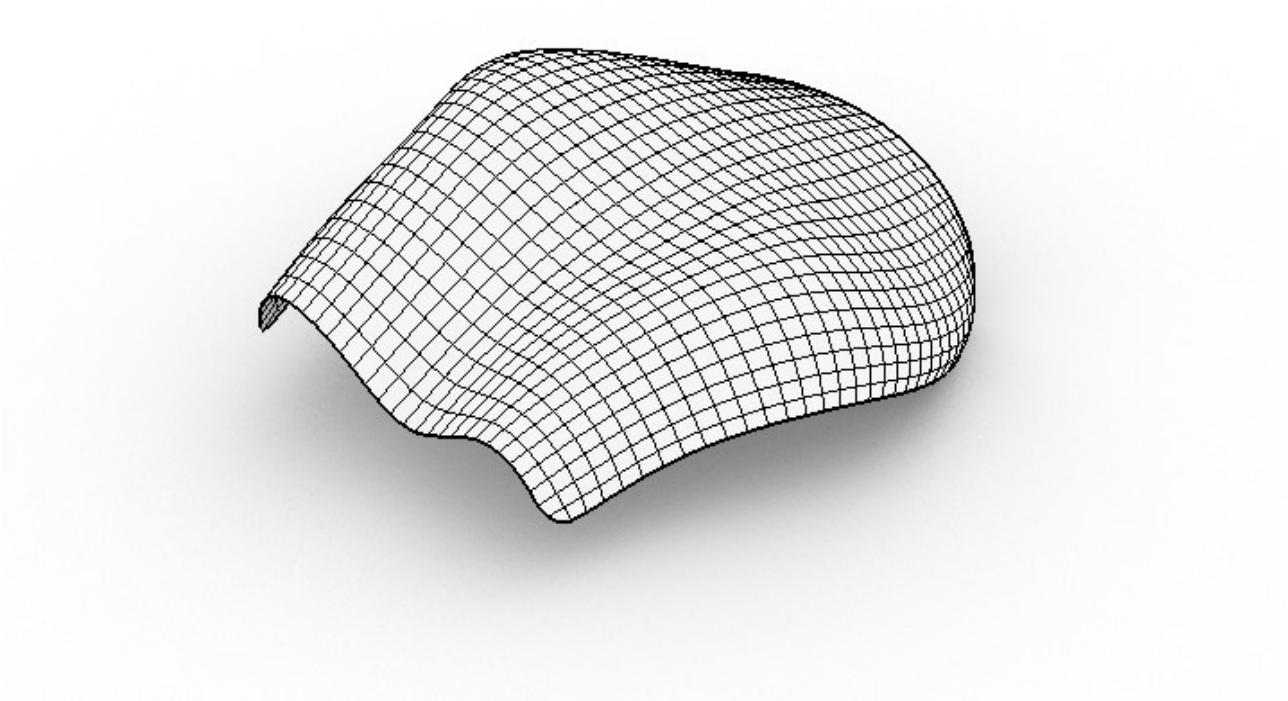
5. 现在你有一个细分网格用来继续下一步的工作，你需要删除原始的粗糙网格。 **强调:** 如果你跳过这步，后面你会得到一些意外的结果！在第 9 章会详细讲解。

6. 在你看到 EvoluteTools 魔术般的优化之前你还需要做一些设置，你需要设置优化参数。首先把所有优化值还原为默认设置，点击图标  或输入 etOptionsReset 指令。优化选项设置中除了 Surface Closenes 与 Curve Closeness 之外的所有默认值都为 0，关于更多的优化选项设置介绍请查看第 8 章的内容。点击按钮  或输入 etOptionsImprotance 指令，然后选择 FairnessCurvture 选项或敲入 A，然后输入 0.3， 再回车两次。

7. 下面是第一次见证 EvoluteTools 奇迹的时刻！将鼠标移至按钮  ，然后点击之，看到奇迹了吗？不会这么快就看到你要的结果了吧？再次点击！实际上你可以继续点击直到你认为面板布局改善到最好或是指令行出现 *“Optimization converged, no more changes were done to the mesh”* 的提示为止。* 你最终的结果应该看起来和下面的图接近。

* 重复点击的原因是 Optimizations 指令仅每次都在一个限制的迭代次数内运行，允许[在 优化切换设置，默认设置为 1]，这样会让你在每一步或多步优化时感觉到网格的变化。如果得到了你想要的优化结果，可以停止优化，反

稍稍点击鼠标，数分钟之内就能得到比较好的结果。

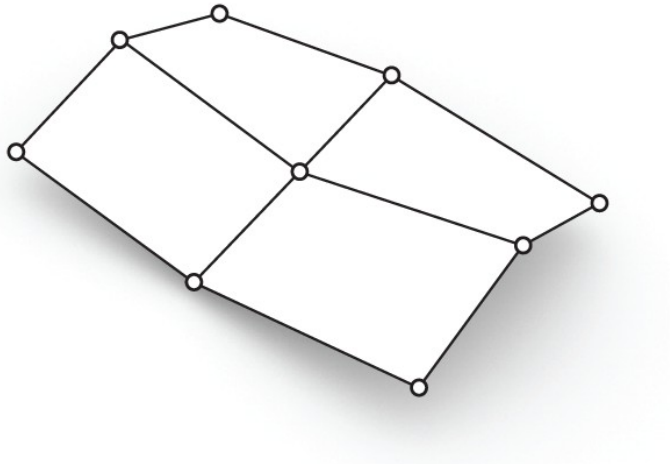


之你可以修改 Optimizations 选项继续优化。

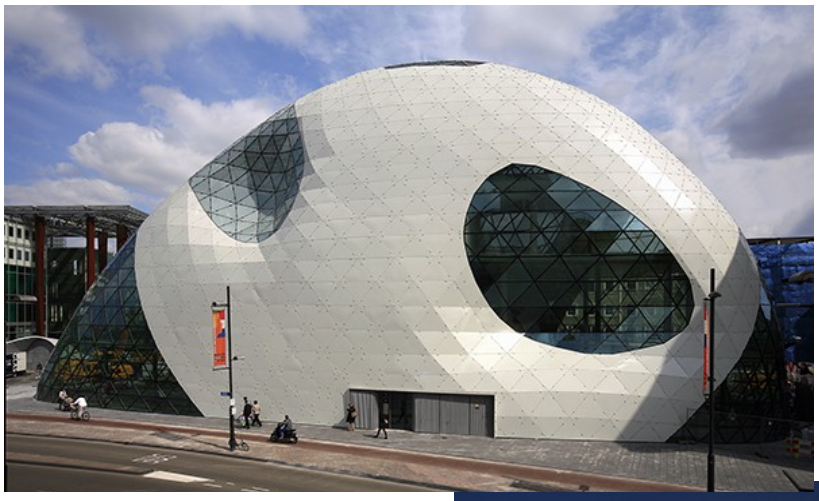
5. 网格基础知识

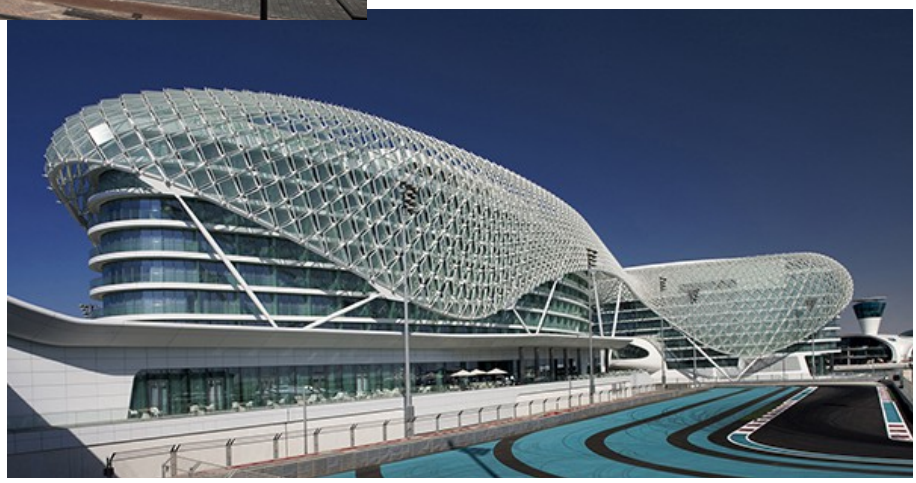
插入这个章节是让你了解到什么是网格，为何 EvolutTools 会把网格作为面板布局与优化的对象。如果你对网格已经非常熟悉或向返回到实例练习章节，你可以跳过这个章节，浏览下一个关与创建粗糙网格的章节。如果你想深入了解网格的处理方式，我强烈推荐由 Pottmann、Asperl、Hofer 与 Kilian 合著的书籍 *Architectural Geometry*。

在大部分的术语中， 网格是一组相互群组连接的点（顶点）来定义一个多边形（数块面），最典型的就是多边形。例如三角面或四边面，这些面都是在共边处相互连接，例如右图就是一个典型的四边网格面。



这不仅是学术问题，把网格与建筑关联在一起并没有想象的那么复杂，顶点、边线与面可以在建筑中表现为节点、梁与面板而组成一个建筑的表皮。像网格一样的结果就会变得更加简单。下面这些建筑就是使用 Evolute 来优化其建筑外表皮结构。



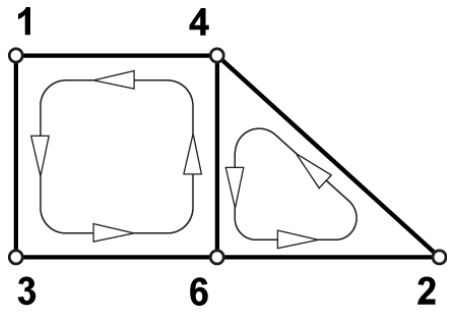


图片. 5–一组像网格一样的建筑。
左图: *The Blob by Massimiliano Fuksas. Eindhoven, Netherlands*

下图: *Yas Island Marina Hotel by Asymptote Architecture, Abu Dhabi, United Arab Emirates*
图片由 *Waagner Biro* 提供

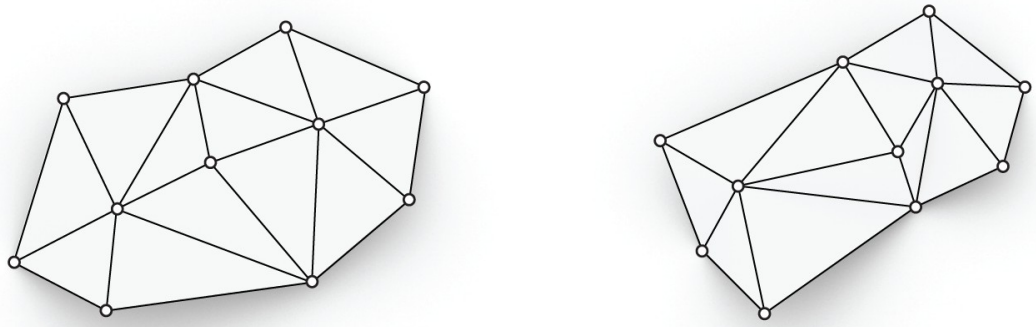
5.1 连通性与几何性

每个网格都有两个基本的属性：连通性与几何性，连通性是描叙网格顶点是如何连接成网格片面的边线，每一个顶点都有一个编号，且这些编号都被收集在定义多边形片面的设置中。[如下图所示]

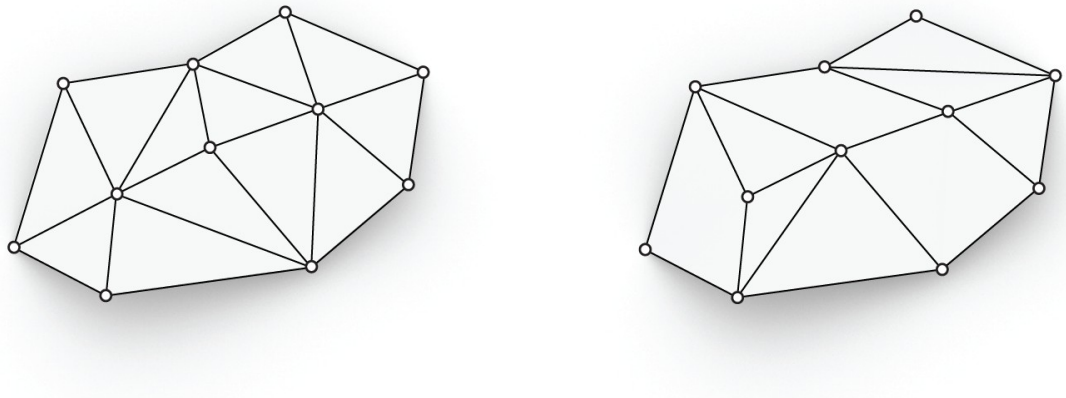


图片 ***6.*** - 片面的信息 *(f 1 3 6 4)* 与 *(f 2 4 6)* 告诉我们:一个四边形面是由编号为 *1*、*3*、*6* 与 *4* 的顶点来定义（逆时针方向），旁边的三角形面是由编号为 *2*、*4*、*6* 的顶点沿着由顶点 *4* 与 *6* 构成的公共边缘来定义。

一个网格的几何是由空间中的顶点来定位，或者说是由顶点的坐标值来定位。在 EvoluteTools 的优化引擎也是基于这样的属性，根据用户设置的规则与设置定义参数不断的来回移动顶点，但并没有改变顶点间的连接性。 另一方面，不同的细分规则常常会改变一个网格的连通性与几何性。为了让你了解这是两个相互独立的属性，请看下面的两组不同的网格面。第一组具有相同的连通性但几何性完全不一样。



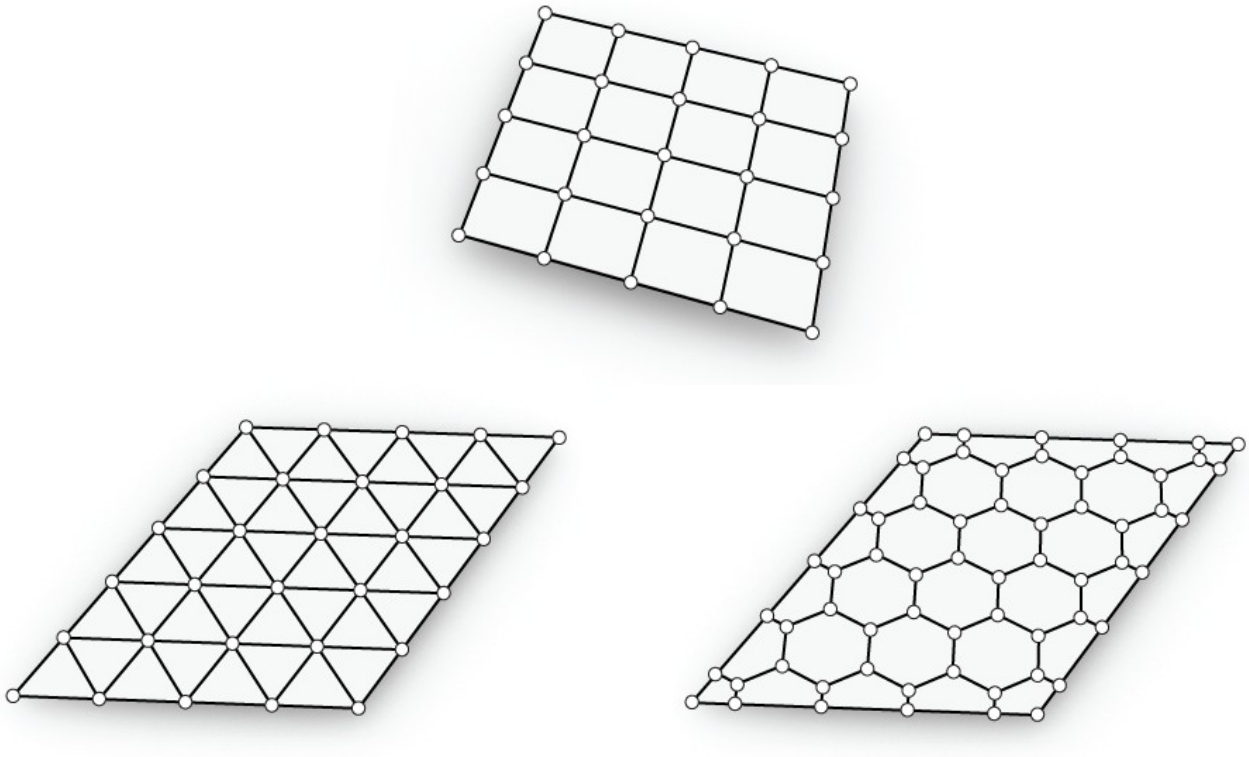
第二组具有相同的几何性，但连接性完全不一样。



大多数的建筑程序都期望所有的面板形状简单且有相同的连接性，但无法保证每个网格都是这样。如果你

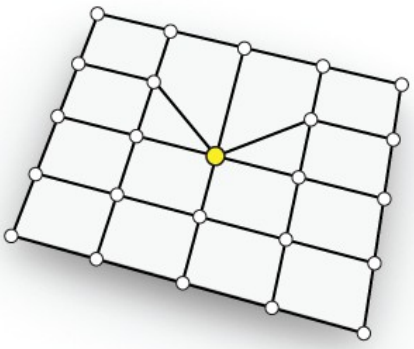
有在方形的面板外创建一个弯曲的栅格，偶尔在五角形的地方会非常的靠外。同样，通常的每一个节点都是由 4 根梁交汇在一起，一旦是 5 根就会打破结构的均匀。

当利用网格来处理的时候有一个技术术语值得提及：这个术语叫“valence(维度)”，用来描叙一个顶点有多少个边线交汇在一起。一张图片胜过千言万语，让我们看下面的最常见的三种网格与顶点维度。



顶部的网格是一个规则的四边网格面，内部所有的顶点都有 4 个一致的边线，因此他们的维度为 4，边缘的维度为 3 且角落的维度为 2。左下图中的网格是一个规则的三角网格面，内部所有的顶点维度都为 6，边缘顶点维度为 4，角落顶点的维度为 2 或 3。右下图是一个规则的六边形网格，所有内部的顶点维度都为 3，边缘与角落顶点的维度依据其外围形状所决定，这张图所示的维度为 3 与 2。

内部顶点维度与常规值不一样（例如维度 5 出现在四边网格的内部维度 4 中或维度 4 出现在三角网格维度 6 中）通常都会给给出一系列的名字，irregular vertices、extraordinary vertices、或 singularities。（查看右边的图片）值得注意的是出现这些会影响美观，但某些场合有无可避免。在那里放置 singularities 对于曲面平直化处理过程尤为重要，我们会在后面的章节详细讲解。



现在为止，这些基础知识已经完全足够。下面的章节，使用这些知识可以让你创建出非常漂亮的基础网格面，这对于最终的面板布局至关重要。

6. 粗糙网格

在你创建与设置参考曲面之后，第一个真正的挑战是创建一个后续能细分与优化的粗糙网格，可能你和我一样，急切向往后续的细分与优化步骤，但花费一些时间且仔细的考虑粗糙网格对最终成品所带来的影响是非常值得的。

6.1 连通性&奇点

细分最大的影响是继承连通性与奇点位置，因为你在后面的步骤细分网格，对一个规则网格会得到漂亮的细分，新的细分网格内部的顶点也会有一个一致的、规则的维度。如果是不规则的网格细分后也是一样，会得到不规则的片面或是不规则维度的顶点。[关于网格的连通性与顶点维度请浏览章节 5 - 网格基础知识]简洁的粗糙网格，你最终的细分也会继承这样简洁的特征，结果的好与会与你最初的粗糙网格至关重要。

为了向你证明这一点，让我们来回顾一下 4.1 章中的第一个范例。

6.1.1 范例 2

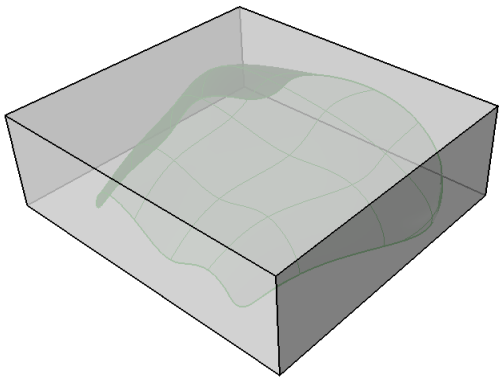
1. 再次开启之前相同的范例文件 [Primer_Example1]，我们设置和第一个范例的第一步一样设置相同的参考对象。

2. 在这个范例中我们会设置一个不同的粗糙网格，选择 Rhino 网格工具中的 Box 工具  ，点击按钮或是执行 MeshBox 指令。

   a. 在命令行做相同的设置的 XFaces、YFaces 与 ZFacese 的设置，都为 1，如下所示。

```
First corner of base ( Diagonal  3Point  Vertical  Center  XFaces=1  YFaces=1  ZFaces=1 ):
```

   b. 下一步，从顶视图选择两个角落点来创建一个方向的薄荷参考几何，然后点第三点确定高度，最终你会得到一个尺寸大概为 50 x 50 x 20m 对象[如下图所示]



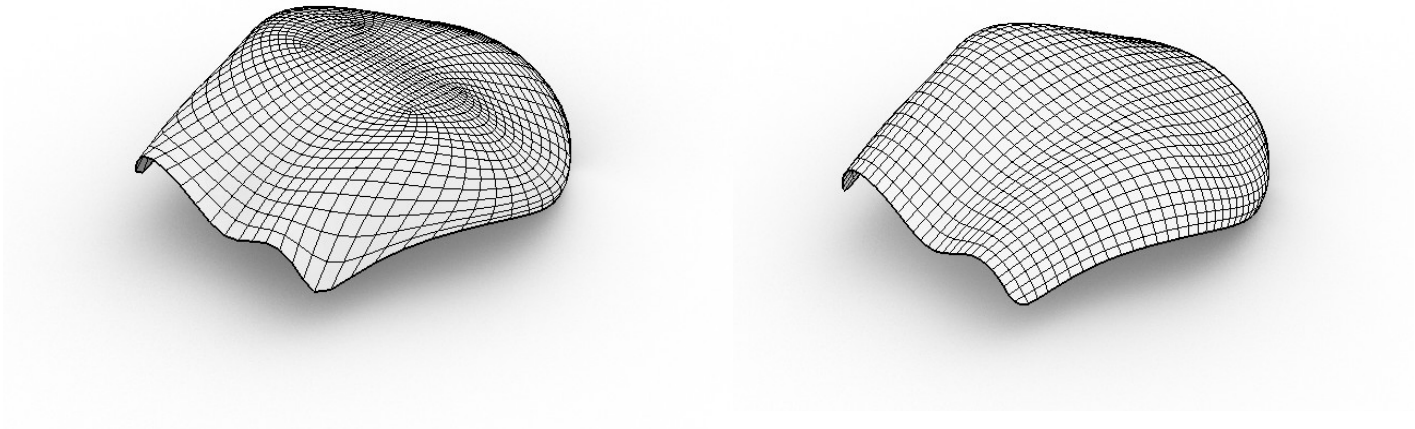
   c. 这样看起来不错，但和我们的参考物体形状并不一致，我们需要删除这个方块的底部。

   从 EvoluteTools 工具列中点击删除网格按钮：  或是在指令行执行 etMeshDeleteFace 指令，选择这个网格方块底部的块面然后回车。

   d. 当你创建或编辑粗糙网格之后尽可能的使用 etWeld 指令  以确保所有的网格足够整洁，

以便后续 EvoluteTools 的进一步作业。

3. 后续的步骤和范例 1 的处理过程完全一样，最后细分与优化的结果如下图中的 Mesh A 所示，与之前由一个单一的网格面得到的结果对比一下。



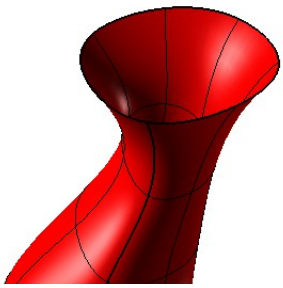
图片.7 - 左图：范例 *2* 的结果，命名为 *Mseh A*， 右图：范例 *1* 的结果，命名为 *Mesh B*。

请注意 Mesh A 中的面板是如何从位于顶部较小的面板群辐射出来的，然而 Mesh B 中的面板反而更加的整齐，请仔细查看顶部的网格。你有看到 4 个奇点吗？在最终的网格中出现这 4 个奇点绝非偶然，他们是从原始的粗糙网格方块继承而来。请注意方块的顶点（位于三块面的交汇处）维度都为 3，而通常在四边形网格中顶点维度都为 4。

这个范例中的粗糙网格违法了精简原则增加了不必要的网格面且产生了奇点，但是要讲的是，越是复杂的参考曲面其边缘也会随之增加，这时候使用一个单一的网格作为粗糙网格会无法作业（例如一个没有边缘的封闭几何体），下面我们快速举例说明：

6.1.2 范例 3

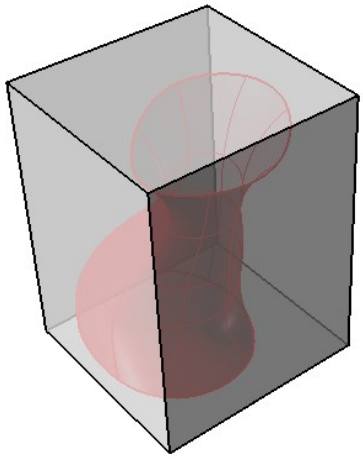
这个范例中你会平直化一个柱状的曲面且会让你了解为何需要一个更加完整的粗糙网格而不是和使用单一网格面。



1. 打开范例文件 Primer_Example3，包含一个新的造型，然后将它设置为参考曲面。

2. 接下来和范例 1 的步骤 3-7 一样的话，你会发现无法下一步的作业。

我们这次使用一个网格方块作为粗糙网格。

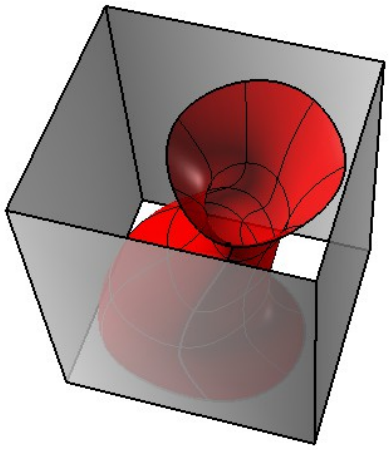


3. 删除你当前的作业网格。

4. 和范例 2 一样，使用 Rhino Mesh Box 工具来创建粗糙网格。

a. 在提示栏，确保相同设置的 XFaces、YFaces 与 ZFacese，都为 1。

b. 下一步，从顶视图选择两个角落点来创建一个接近参考几何的矩形，然后点第三点确定高度，最终你会得到一个尺寸大概为 60 x 70 x 80m 对象[如下图所示]



c. 看起来不错，但我们的参考形状并不是一个封闭的曲面，这次你需要删除这个方块的顶部与底部的网格面。

d. 然后对开发的网格方块使用 etWeld 指令  。

e. 结果如下右图所示。

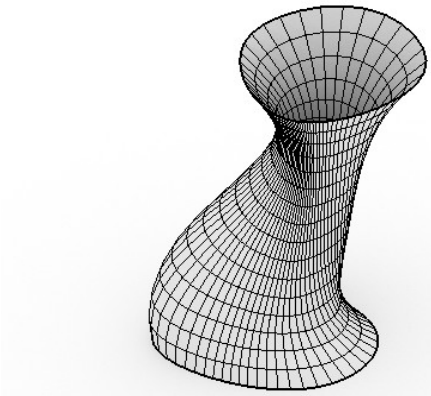
5. 完成粗糙网格，然后使用 Catmull-Clark 细分规则对其进行 4 次细分。

6. 删除粗糙网格。

7. 这次的优化选项和上一个范例稍稍有一些不一样，回复默认设置然后设置 FairnessSpring 为 0.1，FairnessCurvature 为 0.3 与 FairnesscurvatureVariation 为 0.5，* 剩下的选项保持默认设置即可。

```
Importance ( SurfaceCloseness=1  CurveCloseness=1  OriginalCloseness=0  FairnessSprings=0.1
FairnessCurvature=0.3  FairnessCurvatureVariation=0.5 … )
```

8. 现在运行 etOptimize 指令  直到你看到一个比较好的网格分布，看起来会比第一次尝试使用一个单一网格面作为粗糙网格的效果要很多。



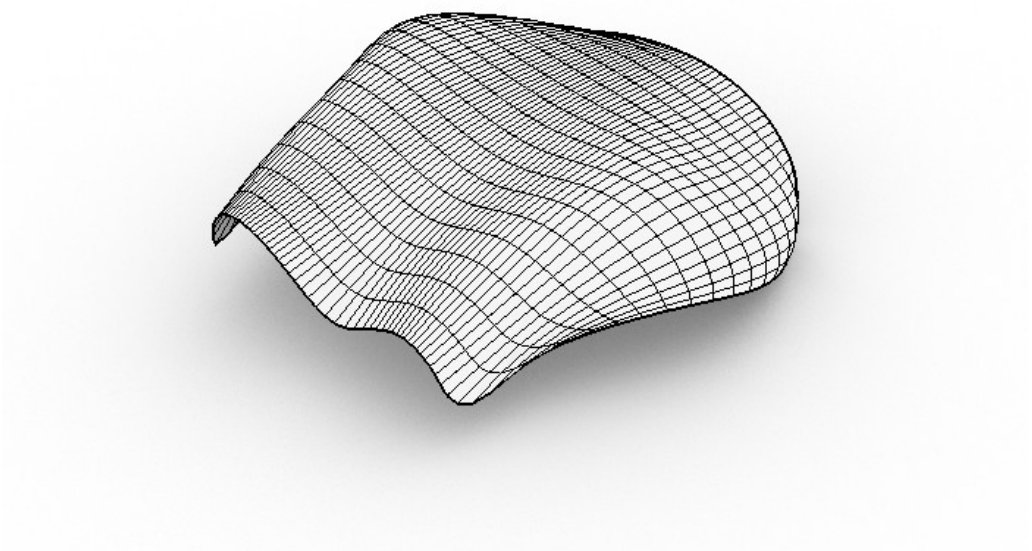
6.2 最终面板属性

除了连通性之外，由粗糙网格细分而来的最终网格也会继承粗糙网格的主要面板属性，换句话来说，如果粗糙网格是由一些细长的矩形构成（例如比较大的长宽比例），最终的细分网格也会是由一些细长的矩形网格组成（当然这和所使用的细分规则相关，可能并不是全部为矩形）。下面这个范例会示范这一属性如何出现在最终的网格。

* 想了解为何要设置这些整顺选项请浏览 8 - 优化篇了解详情。

6.2.1 范例 4

1. 开启文件名为 [Primer_Example4] 的文件，你会看到和第一个范例相同的造型。
2. 接下来和第 1 个范例的第 1 步一样，设置你的参考物体。
3. 在参考物体的上面已经有一个网格面板，将使用这个网格作为粗糙网格，注意这个网格由 4 块长方形网格面组成。
4. 后面的步骤参考范例 1 的第三步起的内容，最后结果如下图所示：



现在明白最终的网格面板是如何保留粗糙网格的细长矩形的比例了吗？

提示：和连通性不一样，在最初的粗糙网格长宽比与最终面板网格的长宽比直接会有一个直接关联，为了检测最终的结果你会时常进行细分与优化的步骤，这样你可以返回修改粗糙网格或透过细分工具简单的调整细分网格，我们会在下面的第 7 - 细分中详细讲解。

6.3 从哪里开始？

现在你已经了解粗糙网格会影响最终的面板网格，并不是所有的粗糙网格都是相同的。当然，你也无法立即判断到底哪种粗糙网格最适合，并没有一个固定的方式与规则，这和用户的经验、项目的需求与对细分与优化工具的掌握程度息息相关。下面的范例会演示什么是创建粗糙网格最关键的步骤。

第 1 步 – 创建

你使用 Rhino 的 3DFace  指令一个个的创建网格能完全控制粗糙网格，适当的配合使用物件锁点，非常容易创建一个接近参考曲面的基础网格。

你有看到，就现在这个范例而言使用 MeshBox 指令非常有用，其实任何 Rhino 的 Mesh Primitives 都可以使用，如果你的参考曲面是开发的，你必须要删除一些面，正如你在范例 范例 3 所做的一样。

警告: 虽然 Rhino 的 Mesh 指令能将曲面转换为网格，但所得到的网格结构很凌乱、不整齐，且太多的细节，不适合当作粗糙网格。

第 2 步 - 组合&焊接

如果你已经创建好粗糙网格，你需要使用 Rhino 的 Join 指令或点击  将它组合在一起，不管你是如何建构粗糙网格，重要的是使用 EvoluteTools 的焊接工具  或敲入 etWeld 指令将其焊接，这一步会清理网格中重复的顶点，为后续的 EvoluteTool 深入操作做准备。

第 3 步 – 设置与/或固定角落顶点

如果你最终的几何上带有锐利的角、点、边缘或是刚好与粗糙网格顶点在同一位置上，这是要你需要将这些顶点设置为角落点且固定，这样让 EvoluteTools 知道这些特别的位置，设置方式请执行 etSetVertexCorner/  与 etSetVertexFix/  ,我们会在 8.4 章中详细讲解。

第 4 步 - 局部的细分（例如 LoopCut、Diagonals、Mesh Cut）

在这一步，你会在粗糙网格的适当位置增加细节或在你开始细分之前修改面板属性。当然你需要一些局部细分工具[详情参考 7.3 章]，知道那些面需要分离且那些是经验问题，很快你就能体会到。

6.3.1 粗糙网格技巧

1. 精简

前面有提及过，再次强调，创建粗糙网格是一个极简主义的训练。

an ancient pond
a frog jumps in
-- plop!

2. 注意奇点

有些情况下网格中的奇点不勉避免，但你可以施加更多的控制，可以将这些点精细的设计在粗糙网格中。请注意方块的角落点全部为奇点。

3. 由 NURBS 控制多边形创建网格

如果你的目标对象是一个控制点很少的单一曲面，你可以使用 Rhino 的 ExtractControlPolygon 指令来快速的创建一个简单的网格，会和你的曲面非常接近。

4. 由 T-Splines 曲面创建网格

如果你有使用使用 T-Splines 插件来创建你的参考曲面，你可以使用 tsMesh 将一个 T-Splines 转换为一个简单的网格面。

7. 细分

细分有两个主要的目的，第一个目的是方便从一个简单的粗糙网格创建一个更小分面的多细节网格，我们在前面的范例中使用细分指令就是这个目的。第二个目的是，它变成了一个设计工具，通过修改网格的连通性提高审美（或造型）属性。

在这个章节我们会全面的介绍 EvoluteTools 中的各种不同的细分工具是如何影响网格的。然后通过两个范例，向你证明细分可以当作一个设计工具。

7.1 选择合适的工具

在 EvoluteTools 中提供两种不同的细分工具，全局细分与局部细分。全局细分工具参考 etSubdivide 指令  中的各种不同选项，用来对整体网格修改。局部细分工具包括  、  、  、  、  允许用户在仅在指定的区域细分网格。

当我们讨论 etSubdivide 指令的各种选项时，会称呼为细分算法。真的是这样，使用程序来一步步的来细分一个网格，每个选项都有一个不同的设置步骤且每个都不一样，但对于一个给定的网格，可以预知其结果。EvoluteTools 包括 11 个整体细分选项（Lite 版本中有 9 个）与 5 个局部细分工具。第一次接触可能会令人头疼，学完这个章节后你可以对每个细分工具自呼其名。

如果下面的这些内容你无法 100%的理解，我建议你使用一个简单的平直网格面来体验这些细分工具，三角面和四边面都可以（还记得 3DFace 工具！），然后慢慢增加几何复杂度。

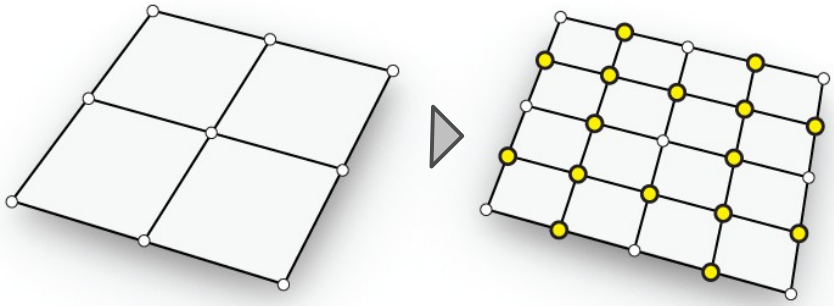
提示：执行完 etSubdivide 指令之后你会发现有两个网格：之前选择的原始网格与新得到的细分网格，他们之间是有关联的，也就是说细分网格的所有顶点都依靠于原始粗糙网格的一个或多个顶点，更多的内容请浏览[章节 9.2 - 逻辑连接]。这一章将集中讲述细分工具的功能。

7.2 全局细分工具

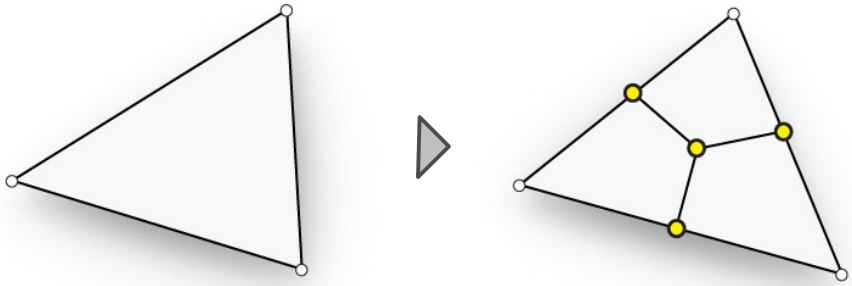
这一节中我会讨论 etSubdivide 指令中的各种细分算法，这些工具会对全局的网格起作用。

7.2.1 Catmull-Clark

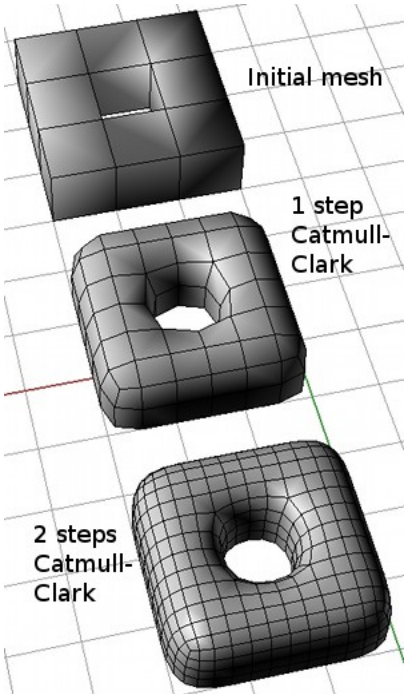
当你对四边形网格进行细分时，CatMull-Clark 将是你的首先细分工具。一般来说，这个算法的工作原理是在每一个块面的中心新增加一个新的顶点，然后在每条边线的中点新增一个顶点，然后透过每个中点与中心点连成新的边线，这样原来每个四边面分成了 4 个更小的四边面。 你无需要了解算法的细节，但也很容易找到丰富的在线资料你可以了解更多算法的相关知识。



Catmull-Clark 算法能对任何的网格起作用，所以并不现在一定是四边面网格，但结果都会输出为四边面网格。例如对一个三边面网格使用这个算法，你也得到 3 个四边面网格。[如下图所示]



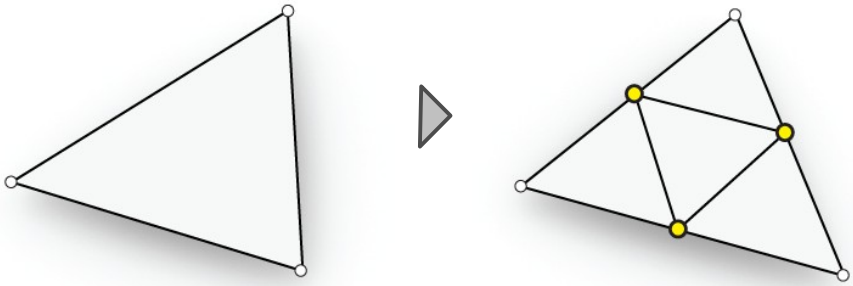
要重点提示的是，当你使用 Catmull-Clark 时你会发现：它不仅仅是新增了顶点、边线与块面，原始的顶点也会基于周围顶点的加权平均值移动，这样进一步的细分可以逐渐的平顺网格。



在这章与下面的环节中，我仅会大概的讲解这些算法是如何处理一个简单的面板网格，对于 3D 造型，它会稍稍复杂一下，对这些功能给你一个初步的了解。

7.2.2 Loop

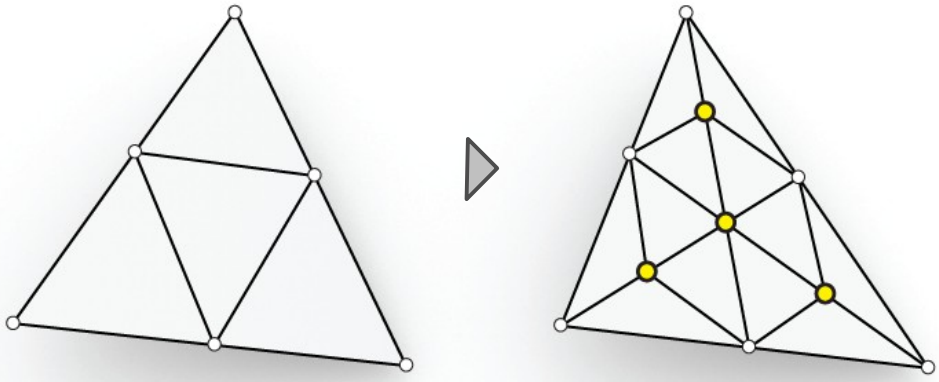
Loop 细分仅仅作业与三角面网格且总是生成三角面网格，这个算法的基本作业方式是在三角形边线的中点位置建立新的顶点，然后将新的 3 个顶点链接起来。这样新创建的三角面内插入原来的三角面，将原来的一个三角面变成 4 个三角面。



7.2.3 Sqrt3 «»

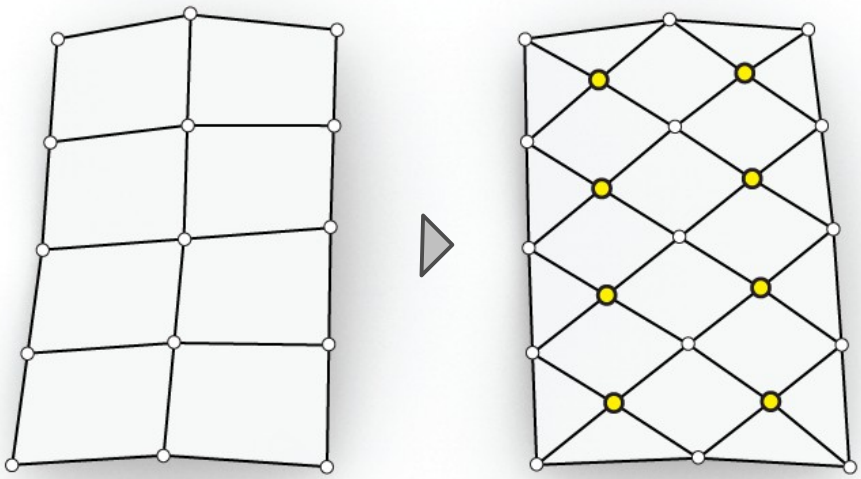
Sqr3 细分算法也是生成三角面网格，和 Loop 的方式不一样，Sqr3 细分算法可以接受任何类型的网格，也就是说没有三角形也会生成三角形网格，但连通性会不整齐，所以最好用在均匀的三角形网格上。

Sqr3 会在每一个块面的中心点建立一个新的顶点，然后将它与当前块面的所有点以及临近面的新顶点连接，然后移除当前的边线。[如下图所示]



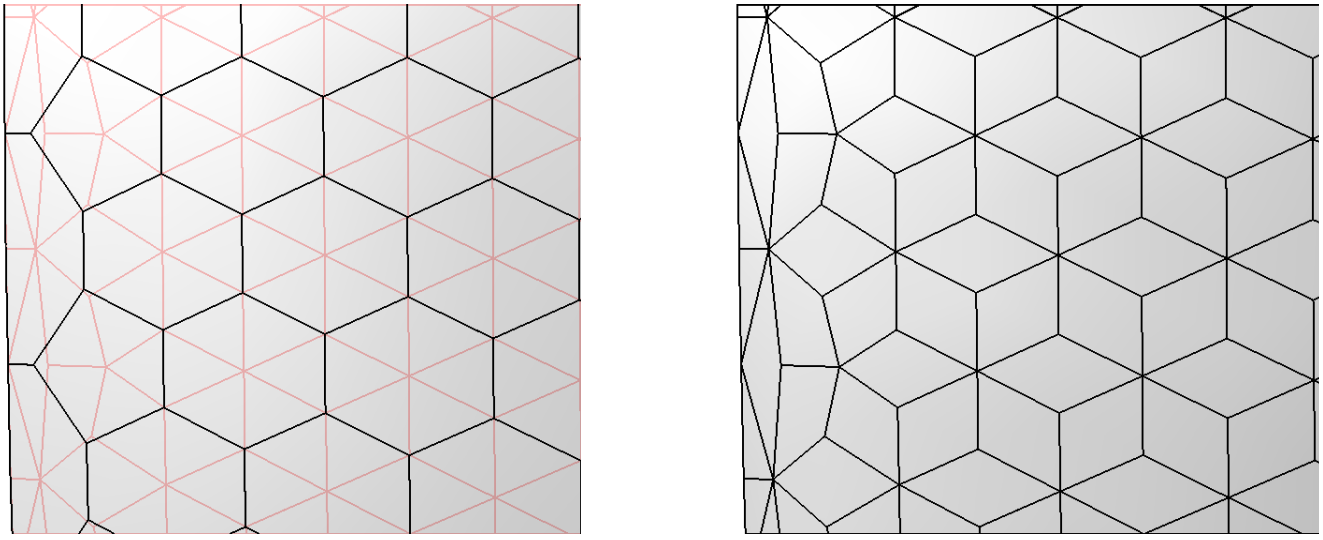
7.2.4 Diagonalize

Diagonalize 细分算法没有任何网格限制，这个算法可以接受任何类型的网格。这算法的本质是在每个块面的中心点建立一个新的顶点，然后将这个新的顶点与当前块面的所有顶点连接，然后删除原始的边线，当你用在一个四边面的时候就会很容易明白为何名为 Diagonalize。[如下图所示]



当你对四边网格面作业的时候，这是个非常好的面板布局转换工具，很容易的将正交架构图案转换为斜肋构架图案。另外可以然给你的面板布局看起来更加的流线。

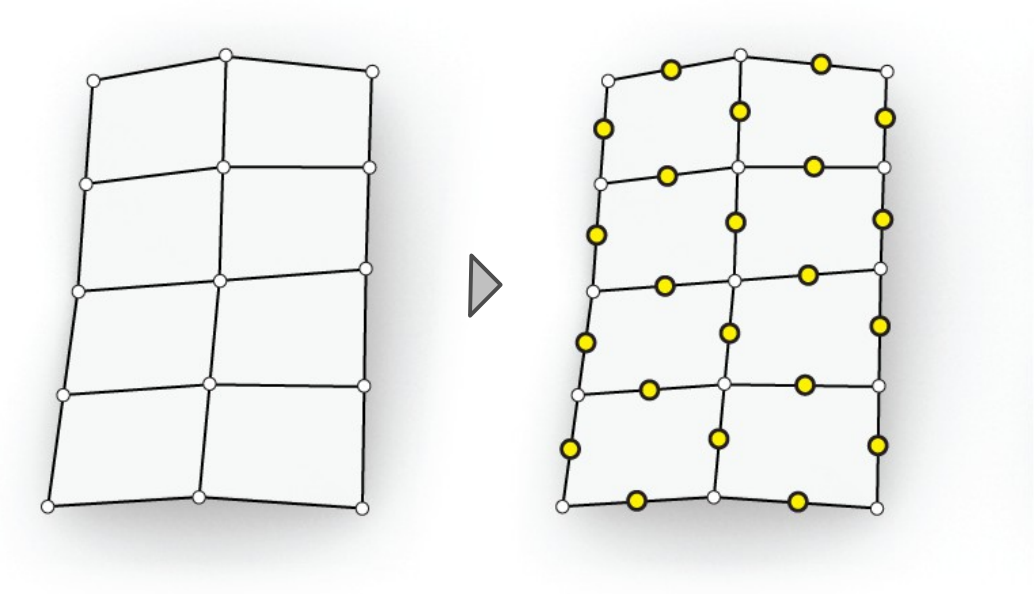
无论输入哪种网格（即使没有四边形与多变网格），最终的结果都会以四边形居多且边缘会生成三角网格，你稍查看上面的图片就会明白原因。每一个非边缘的内部新块面都是由原来的临近块面的两个中心点与前面共边的两个顶点连接构成，及时两个临近的块面为多边形也能建立为四边面。[查看图片]



图片.*8* - 左图：由红色网格 *Diagonalize* 后的 *N-gon* 网格 。右图:从 *N-gon* 网格 *DiaGonalize* 后的网格。

7.2.5 Edge Split «»

Edgesplit 在细分功能中有点无足轻重，它不会创建新的块面或是改变当前顶点的连通性，它仅仅在每个边线上增加一个新的顶点。这样原来网格的边线都变成了两段的多重直线。[如下图所示]



这些额外的顶点可以让网格与目标曲面更加逼近，另外增加的这些顶点与边线段数会彻底的影响后续的细分结果。使用这个细分算法配合其他细分规则可以创建出一些非常有去的面板图案，在后面范例 7 - 组合图案章节中你会看到类似的效果。

7.2.6 Identity «»

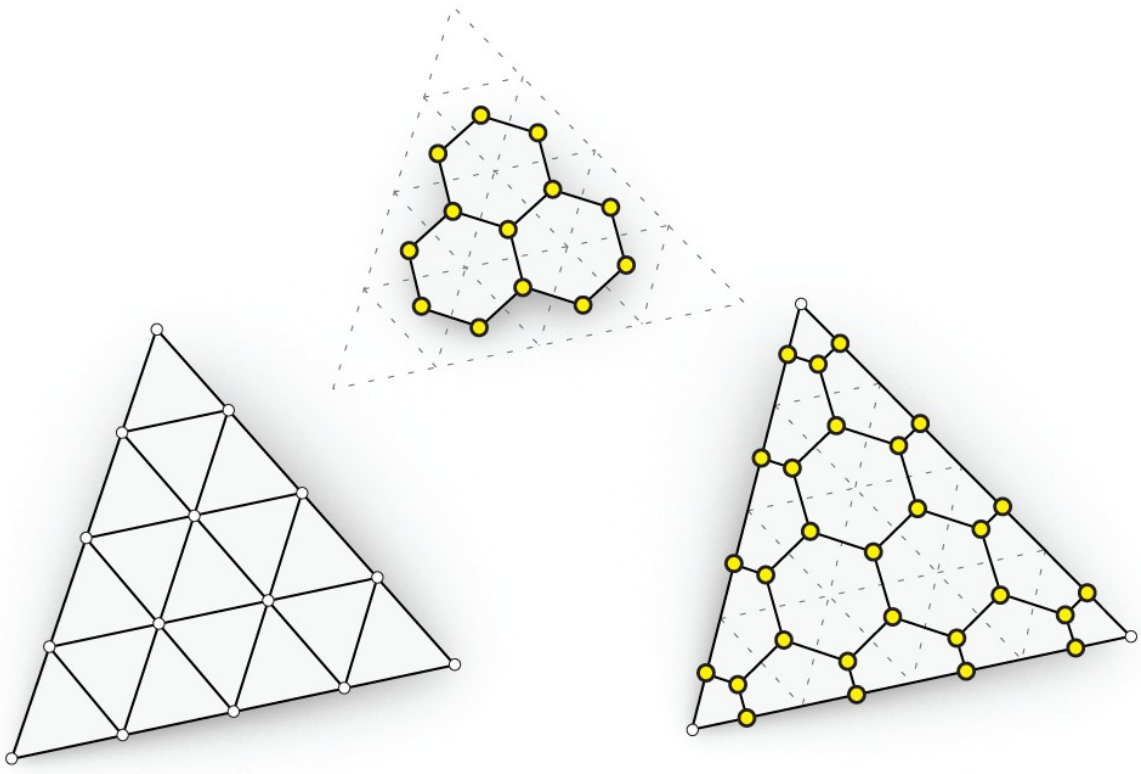
这个细分算法不会执行任何的细分，仅复制原始的网格。第一次使用可能会有些困惑，为何不使用 Rhino 的复制指令而需要使用这个细分方法来复制？为何出现在细分规则下的这个算法不会进行任何的细分？

不同问题相同的答案：使用 Identity 细分工具会让复制的网格与原始网格之间保持一个逻辑的连接，如果你想进行局部细分或是其他的网格编辑（例如删除块面、Loop Cuts 细分等）的同时还要保留原始的网格当作控制曲面，这时候是非常有用的。在 9.2 章中你会发现更多的类似情况。

7.2.7 Dual 与 Dual with Boundary

Dual 与 Dual with Boundary 细分算法最主要的功能是在三角与六边网格之间相互抓换，当然他们也可以对任何网格类型作业且对图案结合也能起到比较好的效果。Dual 细分的基本过程是在块面的中心点新增一个顶点，然后彼此相连的新增顶点连接起来形成一个新的网格。例如是一个四边网格，这结果也会是一个新的四边网格。但如果是一个三角网格就会得到一个六边形网格。在 8.5.2 章中会有更多六边形网格的创建与优化方法。

Dual 与 Dual with Boundary 最大的差别是在边缘是否使用 N-gon 网格来填充。

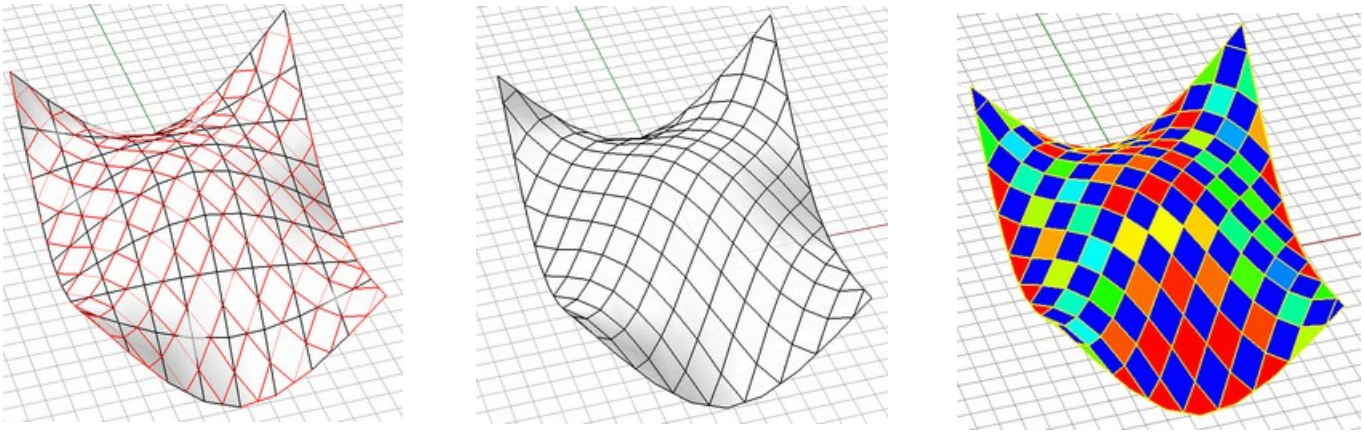


图片.*9 - A dual* 与 *dual with boundary* 细分。**上图：**对三角形网格 *Dual subdivision* 得到 六边形网格。**右下图：***Dual with Boundary* 细分结果。

7.2.8 Dual Edge

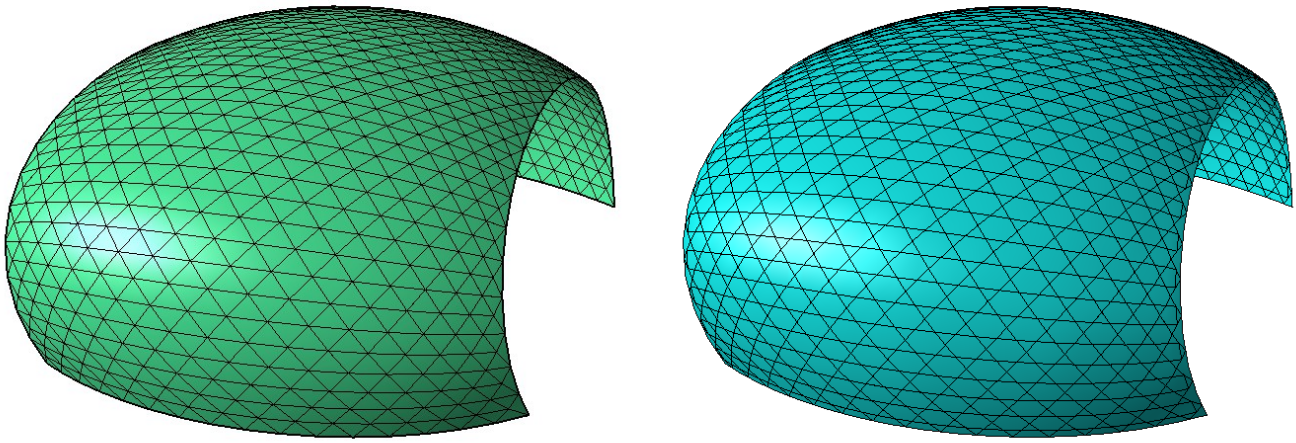
和 Dual 名字很解决但结果却截然不同，这个算法有一些非常独特且有趣的属性。对于四边网格的细分非常有用，最终一半的四边网格会是完全平直的，这是不需要任何优化的，是一个纯粹的几何属性。要提醒的是，其他的非平直面也可以非常快捷的方式拆分成完全平直的三角形网格。 [查看 范例 6 - 快速创建面板布局 章节的介绍]

Dual Edge 算法和其他算法一样开始是在每条边线的中点增加新的顶点，然后使用这新增顶点在网格的没一个块面建立一个内切于原块面内部多边形的新块面，原来的边线会被移除且在两个新的块面间使用 n-gons 填充。



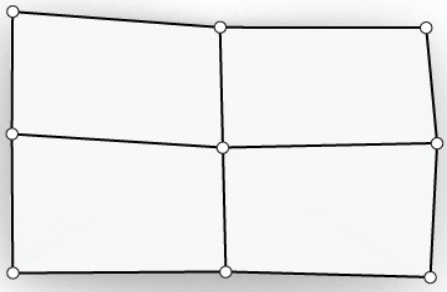
图片.*10*- *A Dual Edge* 细分。 左图：原网格与 *Dual Edge* 后的红色网格。 **中图：**细分后的网格。 **右图：**细分网格平直度检查，深蓝色区域为完全平直。

对三角形的网格执行细分，能得到一个比较好看的六边形与三角形的瓦面图案。

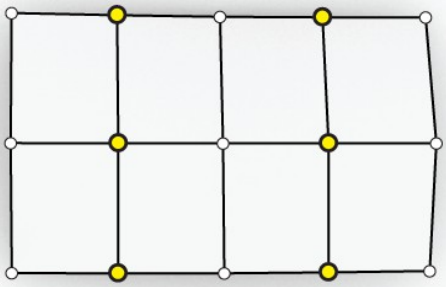


这个功能仅限 EvoluteTools Pro 版。

7.2.9 Strips «»

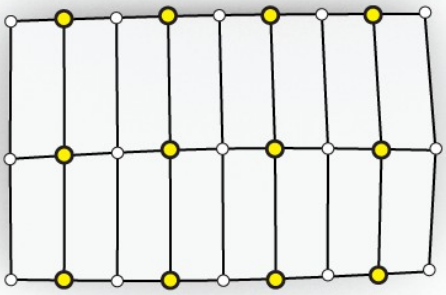


Strip 细分仅对四边网格有效，这个细分算法会把每个网格在指定方向将其边线平分切断，多次重复执行这个细分后的结果会得到近似连续直纹曲面，[提示：并不一定可展开]为了设置执行的方向[例如平切方向]，你必须使用另外的指令来设置，



点击 EvolutTools 工具列中的图标  或是执行 etSetRulingDirection 指令，当你运行这个指令是要求你选择网格上的一条边线作为你所期望的平切执行方向。

技巧: 当创建粗糙网格或是编辑当前的网格的最终细分方法为 Strip 细分时，网格内所有顶点维度必须为 2、4 或 6.

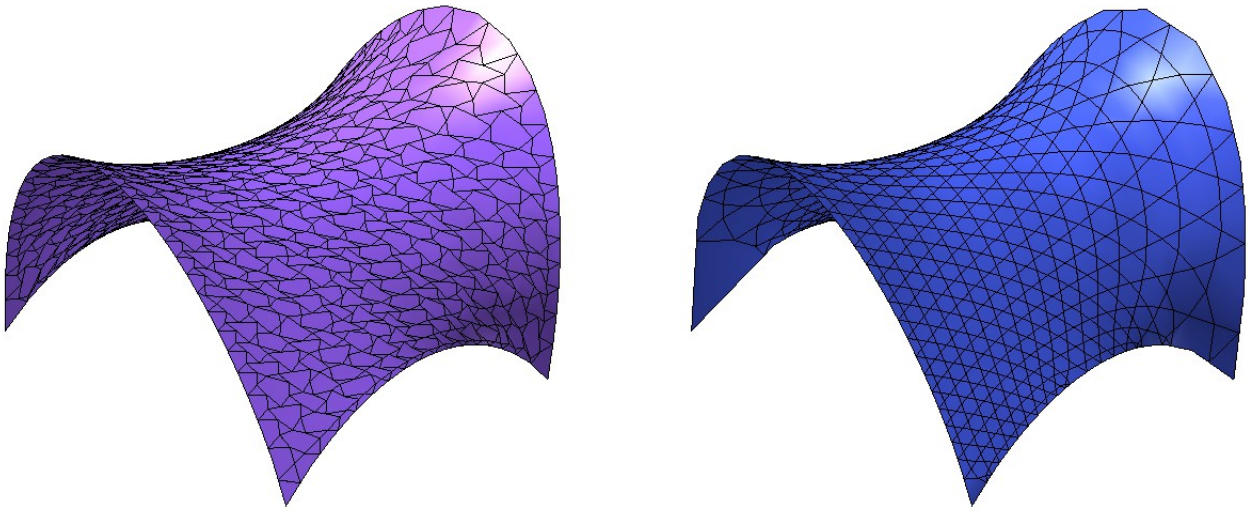


这个功能仅限 EvoluteTools Pro 版。

7.2.10 TriHex «»

TriHex 仅对三角网格有效，最终结果会是混合三角形与六边形的瓦片效果，当对一个三角形网格以 Dual Edge 细分时结果近视，像是“摇晃”后的 Dual Edge 细分曲面。是因为这个算法在当前网格边线上选择了新的点，而不是和 Dual Edge 一样选择中点。TriHex 使用网格的 Ballpacking 属性[查看章节 8.2.7 - Ballpacking «»]来决定新顶点的位置。

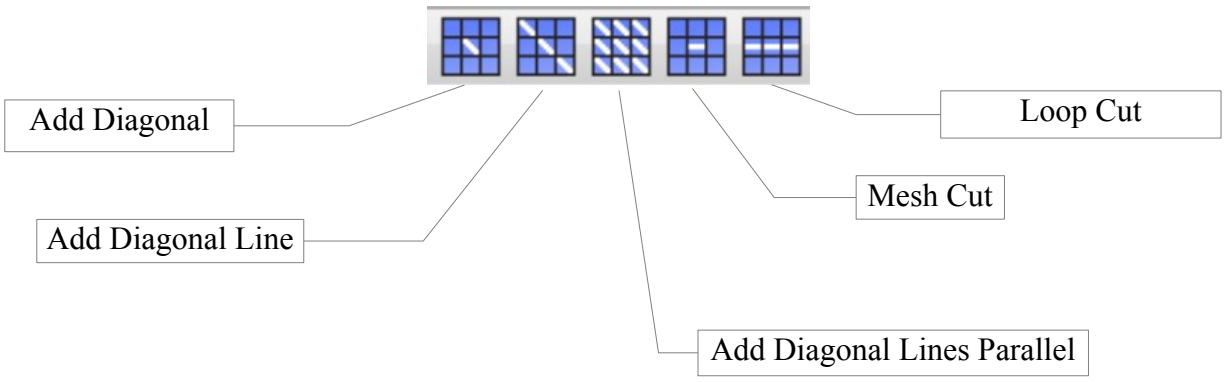
如果有执行过 Ballpacking 优化过的网格使用这个细分算法会得到一个比较“好看”的结果。



图片.*11* - *TriHex* 细分结果。 **左图：** 未 *Ballpacking* 优化网格经 *TriHex* 细分结果，这是一个比较极端的例子。 **右图：** 经 *Ballpacking* 优化网格然后 *TriHex* 细分结果更加规则。

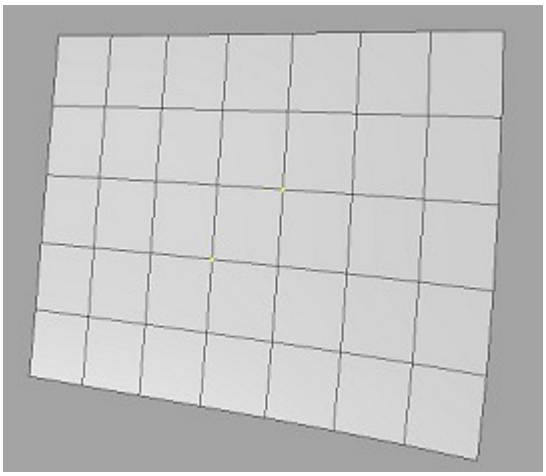
7.3 局部细分工具

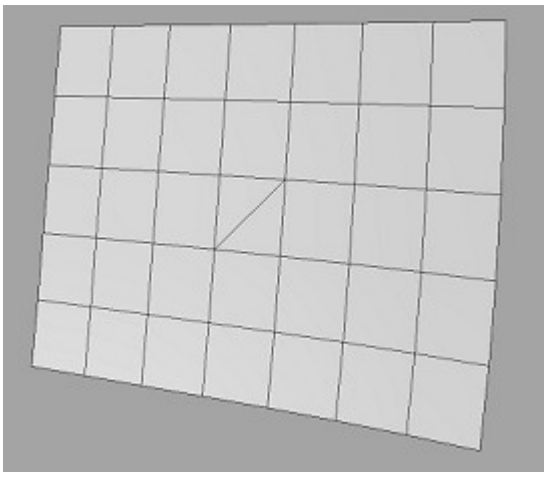
除了全局细分算法之外，还提供 5 个局部细分工具。



7.3.1 AddDiagonal

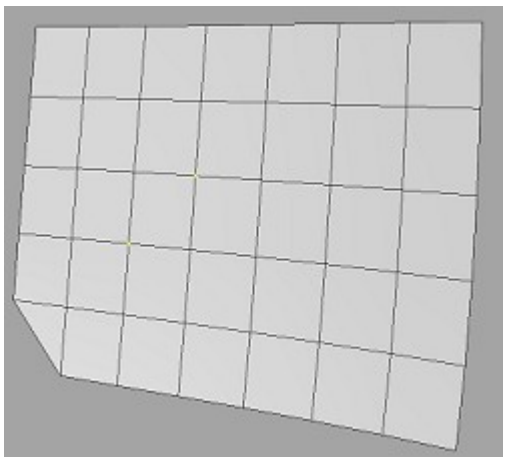
etMeshAddDiagonal 工具会在两个顶点之间新增一条边线将一个四边网格变成两个三角网格，运行是会提示用户选择两个顶点。当选择一个单一四边面网格对角顶点时，会在他们之间增加一条边线。

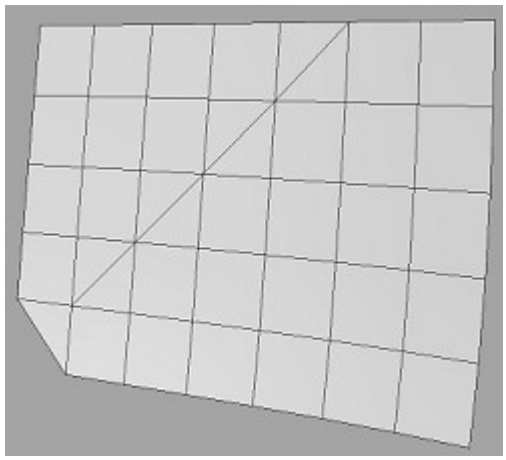




7.3.2 AddDiagonalLine

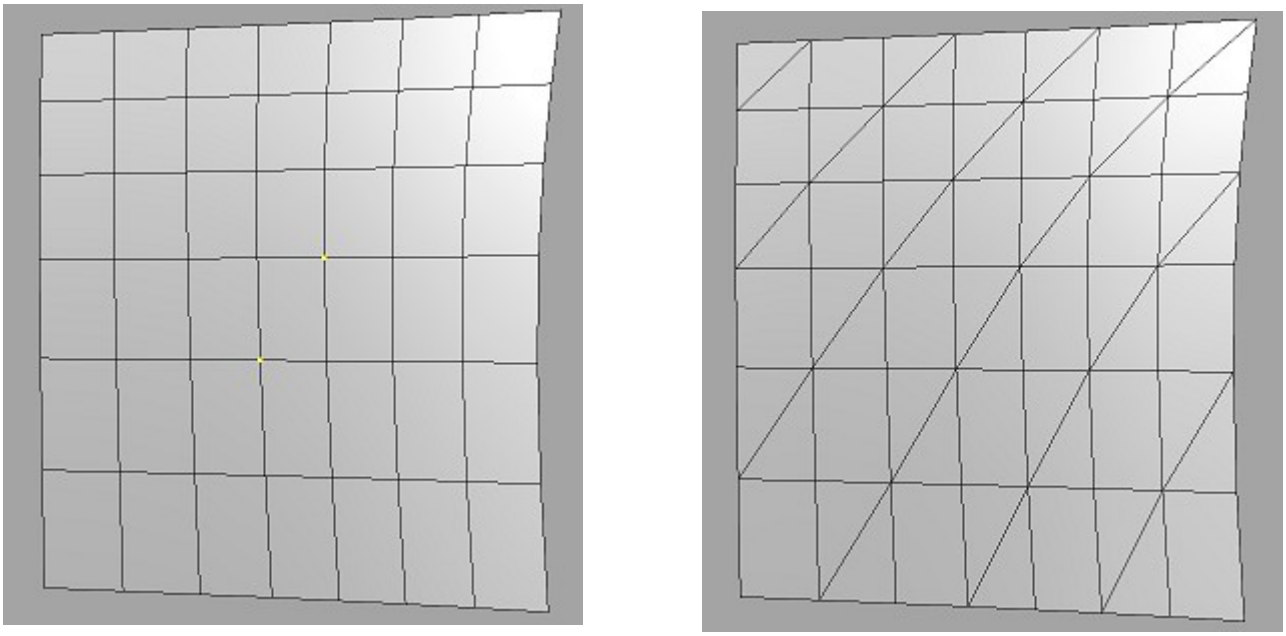
etMeshAddDiagonalLine 的效果与 Add Diagonal 工具执行过程一样,但对角线会在两个防线延续直到碰到非四边网格或是遇到网格边缘框，从而在网格内创建一个新的多重直线。





7.3.3 AddDiagonalLinesParallel

和前面的指令代替一条单一的对角线一样，etMeshAddDiagonalLinesParallel 是在当前整个网格中插入了一组平行的对角线。操作时只要点击两个顶点即可，另外输入一个控制间隔数的整数值。如果顶点是都位于四边网格对角线顶点位置，会在所有的位置插入一条新的边线。这些对角线是连续连续延伸的，出发直到碰到非四边或是网格的边缘。这个只会在网格的所有四边面插入多个平行的对角线，如果有设置“Interval”会在每对对角线间保留所 interval 组四边面，如果 interval 设置为 0，会得到全部为三角形的网格。反之则是三角形与四边面的混合网格。



图片.*12 - Add Diagonal Line Parallel， Interval=1* 的范例。

7.3.4 MeshCut

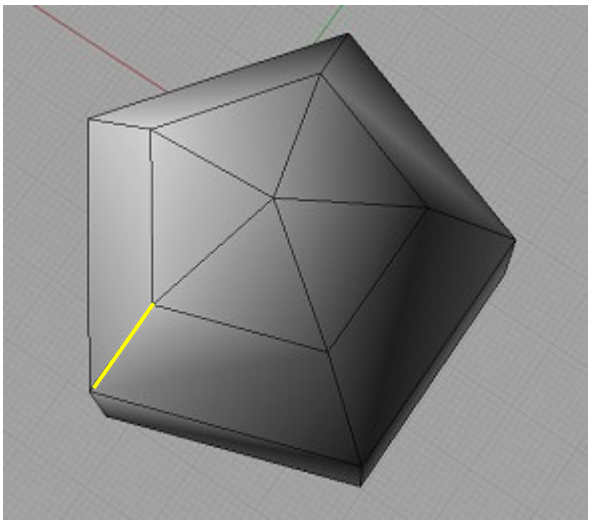
和前面工具不一样，etMeshCut 是在分割一块或是一排网格面，在其边线的中点位置将其均分。执行这个指令只要选择两个网格边线即可，如果选择的两条边线中间包含一排四边形网格，这个指令会对整排网格居中分割。这个指令提供两个操作选项：

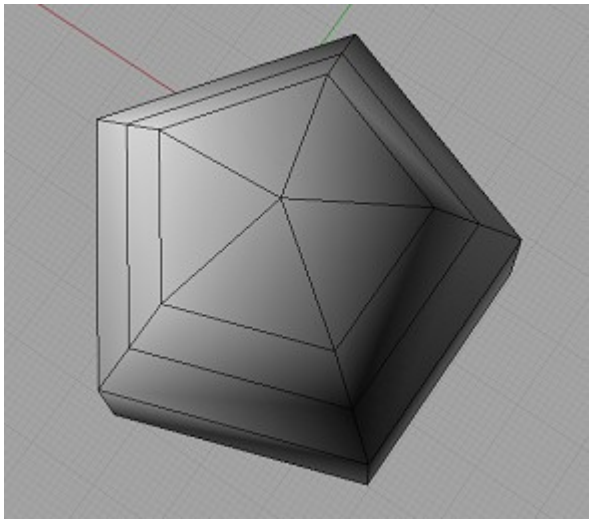
AcceptNgons 的默认设置为 **'false'**,这会造成 EvoluteTools 会在分割的四边形网格面两尾端附近插入三角形。这样做的原因是分割后会让 4 边形上出现 5 个顶点，技术上来说这其实是 5 边线，但 Rhino 并不支持超过 4 个顶点的网格面，这是唯一保证切割后能继续维持为标准的 Rhino 网格面的办法。尽管 EvoluteTools 支持多边形网格的编辑、创建等操作，但在你没有安装 EvoluteTools 时在 Rhino 中导入一个多边形网格会被显示为三角形。

如果 AcceptNgon 设置为 'true',不会出现三角形，这样会导致操作后的多边形中会包含五边形，Rhino 并不支持这种对象且现在的 EvoluteTools 插件还没有提供完全的功能。

7.3.5 Loop Cut

与 etMeshLoopCut 指令与 MeshCut 指令接近，你只需要选择一条边线即可，它会在自动在你所选择的这条边线同排的两次自动居中分割，直到碰到非四边或是网格边缘。可能缺乏 MeshCut 的精准，但可以尽力避免创建 ngons 等相关的问题，我常常使用这个指令来增加网格的密度与细节，是非常便捷的工具。



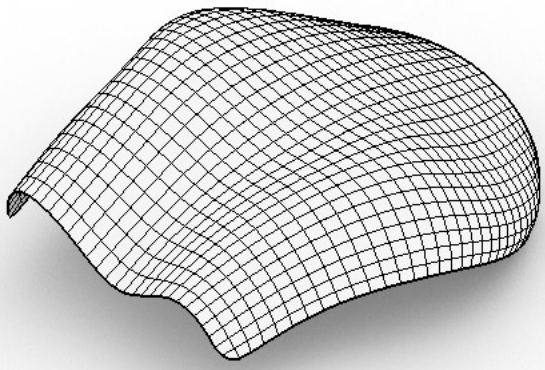


7.4 细分范例

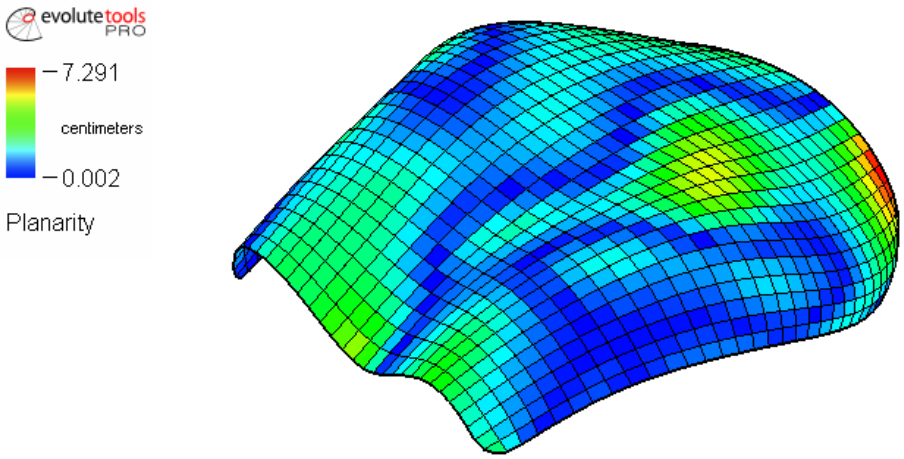
现在我们开始讲解一些细分的实例，第一个范例中，你会使用 DualEdge 细分算法与 Add Parallel Diagonals 工具从一个四边形网格创建一个平面的面板布局，不会用到任何的优化。后面的范例，我们会组合不同的细分工具创建一些有去的面板布局图案。

7.4.1 范例 6 - 快速创建面板布局

1. 打开文件 Primer_Example6，文件内有一个简单的造型，这是从范例 1 优化而来的四边网格，如果你有保存范例 1 的结果，也可以直接从范例 1 开始。



EvoluteTools 提供了一组用于网格检测工具，可以在网格创建与编辑的时候查看网格品质，你可以点击图标  或是在指令行执行 etAnalyzePlanarity 指令，如果你执行前没有选择网格，会提示你选择一个网格。



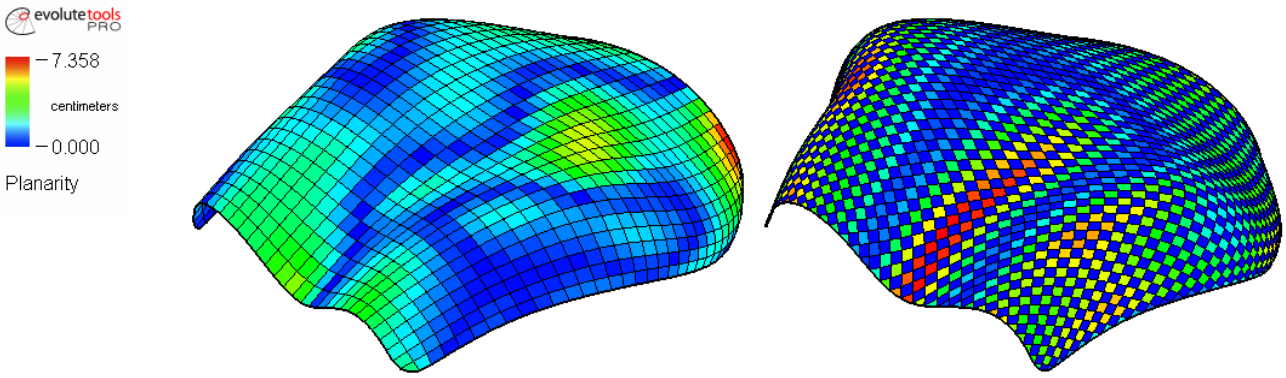
你会看到和上图所示的内容，EvoluteTools 会检测每个网格面有多平直 [详情浏览 8.2.9 章]，网格面的颜色根据弹出框的比例而显示。一般来说红色区域是最不平直的区域，将来加工面板都会比较难。总之这个网格还不算太差，但也有一些区域不太平直。

试试看 DualEdge 细分是否能帮助到我们。

2. 下一步，再次选择网格然后点击细分图标  ，你会得到一个覆盖在原网格上的新网格，在分析显示中还没有立即显示结果。

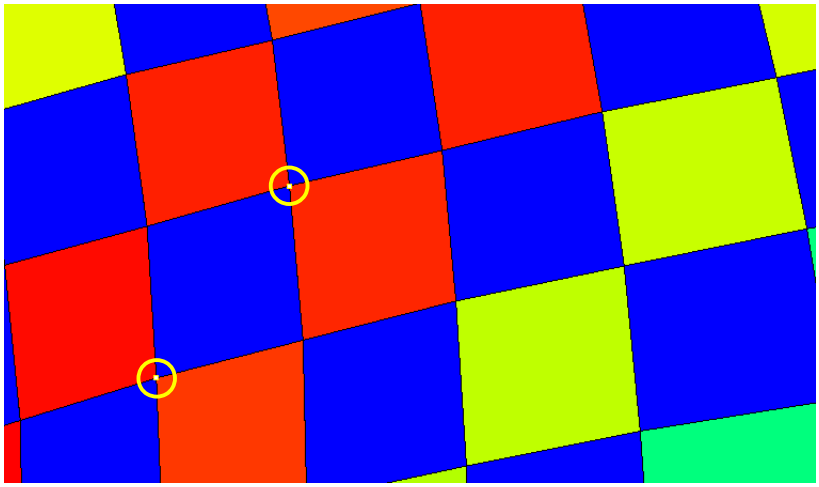
3. 为了两个网格做对比，将前面的网格拖到一边。

4. 然后选择新生成的网格，在网格分析工具对话框中点击 “Add Mesh”按钮，如下图所示：

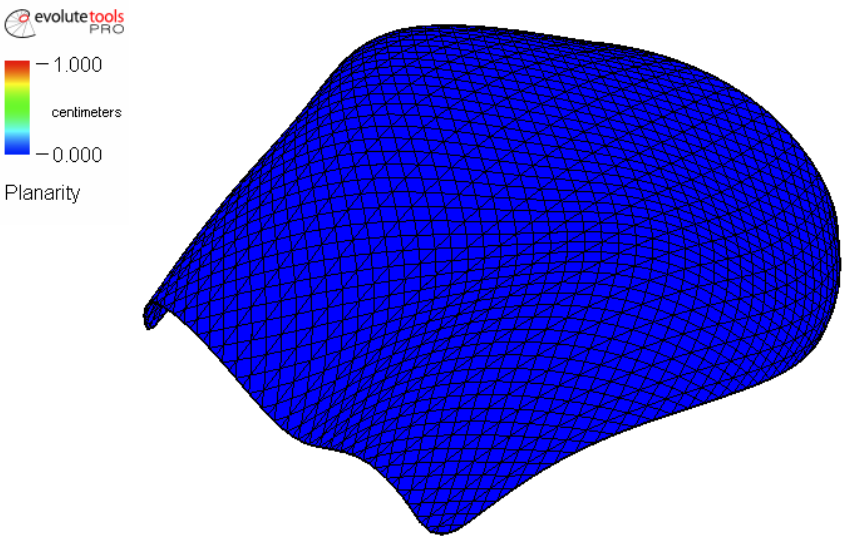


你有注意新网格中深蓝色的交互图案马？深蓝色区域都是绝对平直的面板，其他不太平直的面板我们会在后面修整。

5. 为了让剩下的面变得“平直”你会将它们都切割成三角形，因为三角面一定是平直的（三点共面）。你可以使用 etMeshAddDiagonalinesParallel 指令，点击图标  ，然后选择任何两个非蓝色区域的顶点。



6. 结果如下图所示。



7. 看起来非常漂亮的四边与三角混合网格，如果你认为使用三角网格面板并没有什么技术含量，你需要全部是四边平直网格？或是六边形？在接下来的章节我们会使用优化工具来创建各种类型的平直面板。

7.4.2 范例 7 - 组合图案

1. 打开范例文件 Primer_Example7，注意暂时不要显示名为 Patterns 的图层。

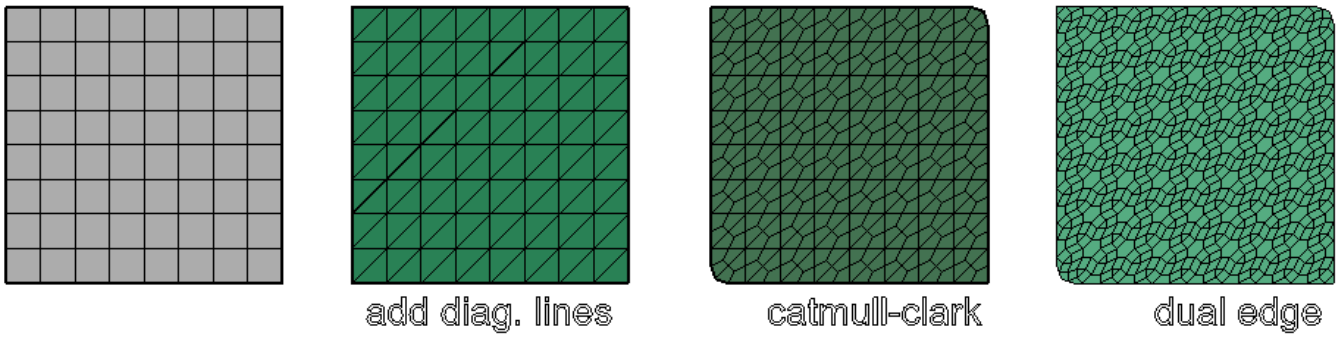
其实显示或隐藏都不太重要，这个图层只是给你一些范例，由不同的细分算法组合而成，我们会逐步的想你介绍如何来创建这些图案。

2. 首先复制这个 8X8 的四边网格，随意放置于一个位置。

3. 执行 etMeshAddDiagonalLinesParallel 指令，且 Interval 设置为 0。这个范例中你创建的对角线方向无关紧要，仅仅只是细化图案。

4. 然后执行 Catmull-Clark 细分，将上一步产生的三角网格全部转换为四边网格。

5. 很有趣， 但还没结束。继续对网格执行 DualEdge 细分， 现在你会看到一个非常酷的混合图案，有四边、三角与六角形。



重要免责声明： 我有鼓励你组合不同的细分算法来创建一些非常奇特的细分图案，但要提醒你的是，Rhino 现在还不支持多重网格（例如非三角形与四边形混合构成多重网格）。EvoluteTools 可以让你在 Rhino 内对多重网格进行创建、编辑等操作，但 Rhino 开启非常复杂的多重网格或是含有较多的 n-gon 网格的多重网格时，可能会出现挂机的现象。因此如果你计划把面板图案设计的很复杂时，请在细分之前**备份你的文件**，我们会在后续的版本中改进这样的问题，但现备份文件是最安全的做法。

8. 优化篇

你可以把这章方到最后（或跳过这章）。在第 4 章的第一个范例，你有看到 EvoluteTools 魔术般的作业方式，当然这是真的，EvoluteTools 不是魔术，因为 EvoluteTools 有一个对网格几何进行迭代变更的优化引擎，持续比较优化的网格与最终目标几何体，更加你设置的优化约束条件持续不断的调整、优化网格结构。

什么使 EvoluteTools 优化方法如此强大以至于可以根据你可以指定任何你想要的约束方式，让解决方案优化排序，你可以在任何时候添加约束而无需重新设计布局。通过控制每一个约束的强度，你可以根据你项目的需求量升定制优化选项。

你可以把优化处理当作一个多方向的拔河游戏，每一个约束都是这个游戏的参与者，你所定义的约束强度值就好比它们在绳子上施加的拉力。可怜的网格会被这些约束条件“拉扯”着，直到优化引擎根据你所设置的强度值找到一最安全的方案为止。使用 EvoluteTools 的真正意义并不不是了解每个约束条件是如何单独的影响网格，而是如何从中找到你想要的合适的平衡。更多的细节，请查看 9 - 如何 充分利用 EvoluteTools 。

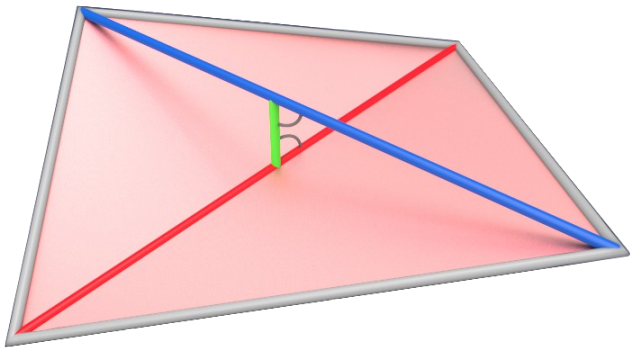
这里有三组工具会影响优化的结果：Optimization Options Importance 或和我一样称之为“全局约束”、优化切换与局部约束。下面我会逐一讨论，实际操作总是组合使用。会有一长串名字古怪、功能复杂的选项。我会尽可能的讲的生动一些，这个章节可能不太适合从头到尾的阅读，反而更加适合参考阅读。

如果你对应用更感兴趣而不是理论，且注意力不集中，我建议你阅读下一章优化范例的优化范例练习，这样让你对优化有一个大概的印象。

8.1 优化简要介绍

阅读完这章之后，你会对我们讨论优化的含义有一个比较好的概念。最后，我会草拟一个简短的、主要的定义，然后再讨论利用 EvoluteTool 如何实现。优化，总的来说就是最大化或最小化一个函数的系统过程，现在明白了吗？

首先，“系统过程”是重复软件处理数码魔法，就 EvoluteTools 而言，仅意味着在每次迭代之间如何改变网格的漂亮逻辑。有一个用于网格的算法且能找到一个“稍胜一筹”的变异，毫无疑问不断重复执行，知道找到一个解决方案。这些对终端用户来说并不重要，真正重要的是如何让我的网格变得“稍胜一筹”？如何实现？



其次，“最大或最小化一个函数”，对于 EvoluteTools 来说这个功能是一个数学的方式来描述你所关注的网格的样子。例如，你想要在一间房中找到最矮的一个人，你最关注这个人的样子：他们的高度。但这并不能整体的描叙一个人，但足以让你找到人群中最矮的人。这是一个非常重要的因素，让优化**可测量且可计量**。例如，无法用数学的方式来描叙一个最好的人，所以利用计算的方式是无法找到这个最好的人（答案往往是：我）

每一个 Optimization Options Importance 对话框都是描绘你网格的一个单一的可测量与可计量的样子，数学的描绘就是函数。具体的例如一个网格的平直度，就是指测量其对角线之间最段的距离。

这个距离是可测量且可计量的，整体平直度函数， P_{total} 表示网格面的数目，N 用来描绘在网格上每一个面的距离的总数。

$$P_{total} = P\ of\ face\ 1 + P\ of\ face\ 2 + P\ of\ face\ 3 + \ldots + P\ of\ face\ N$$

数字越小表示网格越平直，就这个函数而言，“稍胜一筹”表示 P_{total} 中一个更低的数值，这个优化函数将尽可能的最小化它。

前面我有提及过，EvoluteTools 的优化工作就好像一个拔河比赛，因为你会多方面的考虑。你有很多组优化函数，但是如何同时处理这些函数呢？是把你所设置的所有优化值的结果相乘，然后把它们放在一起。这就是一个加权函数。

全部优化函数 = [加权值 1] * [函数 1] + [加权值 2] * [函数 2] + … + [加权值 n] * [函数 n]

如果你把一个优化选项的值设置为 0，表示告诉优化器屏蔽这个优化功能。值设置的越高，对最终优化的结果影响越大。

在下面的章节，我会详细的介绍每个优化选项对与最终优化结果的影响，我会指出哪些是比较重要的地方，我会简单的术语介绍 EvoluteTools 的计算函数。

8.2 优化选项设置 - 全局约束

接下来我会单独的讲解每个优化选项的功能。你有读过且了解到这些功能都是用在一起的，但不会很清楚的确定所需要设置的参数，这些都需要不断的尝试和体验。

点击 EvoluteTools 工具列中的图标  或是在指令行输入 etOptionsImportance，执行优化选项全局设置。

8.2.1 Surface Closeness

大部分情况下，这个选项会是一个标准的设置且默认值设置为 1，这个约束会保证网格顶点尽可能的贴近与参考曲面。这个算法的实际功能是测量网格上所有顶点与参考曲面之间的距离，然后尽可能的最小化这些值。

这是一个非常重要的约束，因为它为顶点的空间定位给出了一个基本内在次序。如果这个值由于其他因素的影响而变得比较低，在网格上会产生出一些不好的脊线与折线。如果要精确的匹配参考曲面，其他的约束参数会比它更重要，我会建议你逐渐的提高其他的约束参数而不是降低这个约束的优化值。

8.2.2 Curve Closeness

这个优化设置会最小化网格边缘顶点与参考面最近边缘的距离值，和 Surface Closeness 的功能接近，这个算法会测量边缘顶点与曲面边缘直接的距离且全部加起来，然后最小化这些值。默认设置为 1。

默认情况，优化器会自动确定网格边缘与参考面[查看 8.3.6 - DefaultBoundaryCloseness]，可是会把网格边缘顶点与参考面直接的距离限定在一个适当的值之内。[查看 8.3.8 – CutOffCurveDistance «»].

对于几何平直度的优化选项有严格的约束（例如， Ballpacking, Ideal EdgeLength）可能会和这个选项相关，或是完全关闭这个选项，它允许网格偏离后溢出参考面边缘以满足其他约束要求，在这些要求都满足之后，另外再增加或修剪网格来更好的匹配边缘网格。

使用 EvoluteTools Pro 可以使用 etSetVertexCurvepoint 指令将网格顶点作为曲线点，且可以使用 etSetReference 将曲线作为参考对象，将内部网格顶点沿着参考曲线对齐。

8.2.3 Original Closeness «»

Original Closeness 最小化网格顶点至其原始位置的距离，换句话说这个优化程序会最小的改变顶点的坐标。

当你需要微调网格时这是一个为你保留最终调整的参数，当你没有参考曲面时用来找形分析也是诶出有用的。

警示：使用 Original Closeness 参数时会把当前选择的网格定点位置作为在你运行 etOtimize 指令时所有顶点的原始位置。重要的是它增加的迭代[查看下面的第 8.3.5 章]次数，会让优化运程序持续运行至收敛，否则最终的效果可能会持续偏移，会与你想要的结果偏离。

8.2.4 Fairness Springs

这是一个非常有趣的用来尽可能最小化边线长度的约束条件，想象这样一个场景如果所有的边线都是弹性的会是怎样，这样就会“紧缩”网格。对于其他的约束条件造成网格多重直线扭在一起的时候这个参数非常的有用。这个参数对最终结果影响很大，因此我会建议你设置的强度值尽可能小。当 Surface closeness 设置为 0 或没有参考曲面的时，可以用这个参数创建极小曲面。

8.2.5 Fairness Curvature

这也是一个你会常常用到的约束参数，它会帮你减少网格中出现的尖锐点现象，给你一个看起来很顺滑的网格。如果把网格看成由多重直线构成的网架，这个参数会让它尽可能的顺直。将而言之，很多时候它会让你的网格看起来更好一些。

想象一下你有一个起伏很大的曲面，且手里握着一个长的绳索，如果要你把这个绳索尽可能的直的排布在曲面上，你会怎么做？开始你肯定会避开最高和最低的起伏点，以及曲率很大的位置。Fairness Curveture 就是这样对所有的网格多重直线执行这样的优化。此优化会大量的移动网格的顶点与多重直线，以便后续

的优化设置。

8.2.6 Fairness Curvature Variation

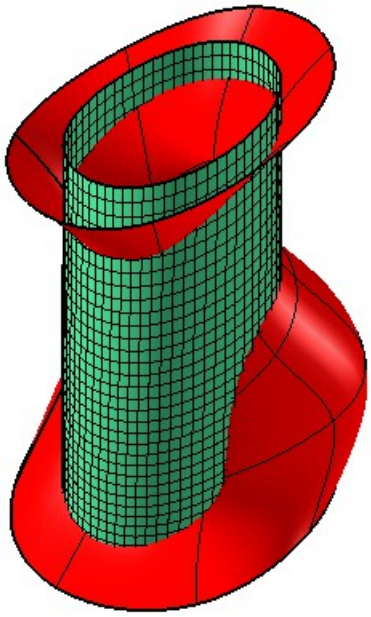
和 Fairness Curvature 尽可能维持网格多重直线更加直（使其曲率为 0）刚好相反，这个优化参数会尽可能的维持其原始曲率-理想圆，“好，”你会对自己说，“这很好，但我什么时候需要用到这个参数？”简单的回答是当你有一个几何形态对于 Fairness Curvature 优化不很好的时候，“什么样子的几何形态呢？”

正如前面提到的，Fairness Curvature 会避开曲率高的区域。比如有一个圆柱造型，会在一个方向有有一个明显的曲率，执行 Fairness Curvature 时会将网格的多重线拉直且把他们移动到不想要的位置。换句话说，Fairness Curvature Variation ，不会太在意弯曲的多重线，它会尽可能的均衡曲率以至更加统一（像一个正圆）。使用这个参数会减少网格的扭曲、扭转与位移。

让我们做一个简单的范例：

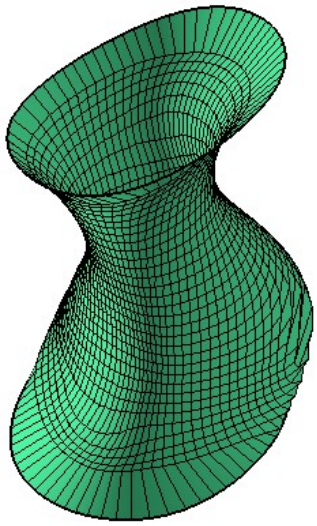
8.2.6.1 *范例*8 - Fairness Curvature 与 Fairness Curvature Variation

打开范例文件 Primer_Example8，包含一个 6.1.2-范例 3 中圆柱的参考造型与一个准备优化调整的细分网格。这个范例中我们会讲解为何要使用与前面不相同的设置参数。

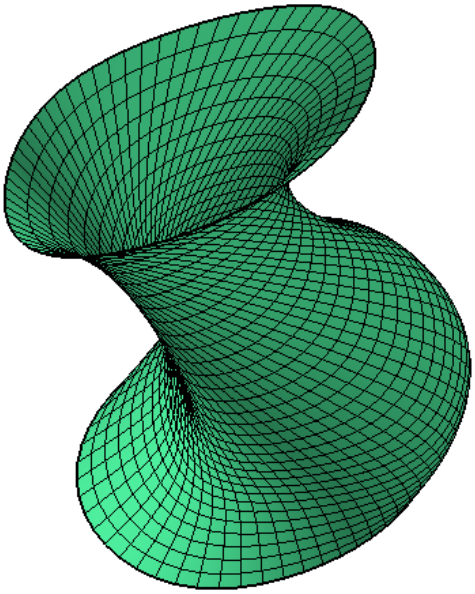


1. 点击  或执行 etOptionsReset 指令恢复默认优化选项。

2. 下一步，点击优化一次，网格会捕捉到参考曲面，效果并不好，你会发现网格上有些扭结与歪斜[看右图]，这是因为我们没有使用 Fairness 参数来顺滑。



3. 将 Fairness Curvature 设置为 1，然后运行 etOptimize。提示：这个值设置的越高，效果越明显。

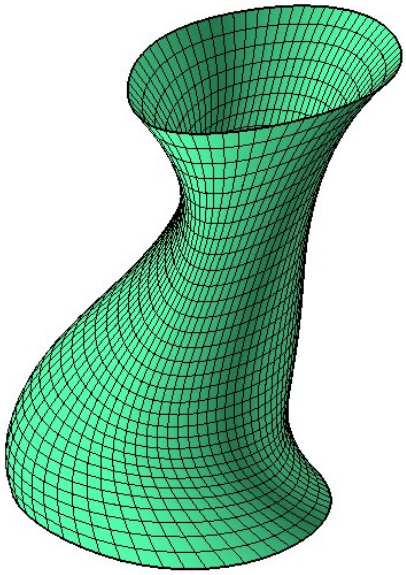


4. 稍后你会注意到网上垂直的多重线开始围绕这个形态扭转[如左图所示]。如果你熟悉双曲结构，你会很明白造成扭曲的原因，如果不清楚，我强烈推荐你 查看 Vladimir Shukhov。

5. 如果扭曲不是你期望的，你需要另外的解决方案，取消这一步的优化，或再次打开原范例文件。

6. 确定你有恢复默认设置，这次设置 Fairness Curvature Variation 为 1，然后运行优化。

7. 最好结果如下图所示，注意已经看不到扭结但还有一些歪斜。当然现在还有一些明显的趋向原造型的弯曲走向，且有一些歪斜的面板。[如下图所示]



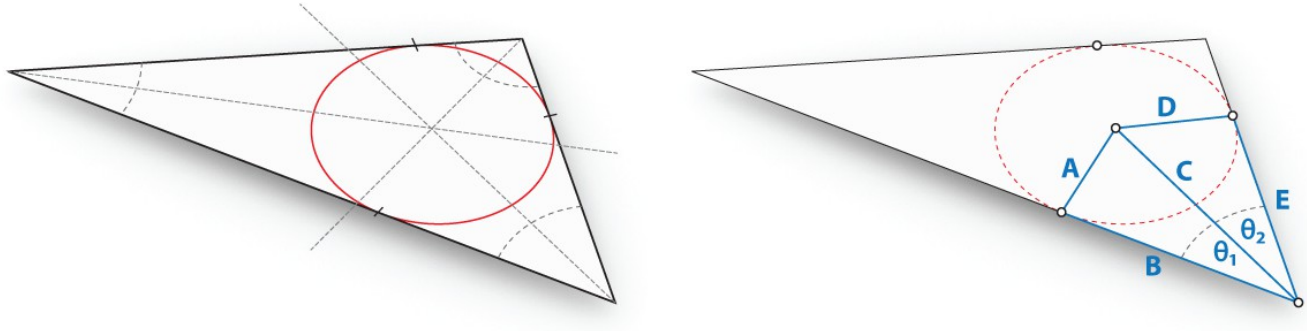
8. 现在稍稍的增加一些 Fairness Spring 优化，如果你设置为 1，它会将明显的弯曲网格拉回，这里建议设置为 0.4。

提示：实际操作中，正顺参数的调整都是不可预知的，完全取决于你曲面属性。最后，送给你一个诀窍，这个诀窍分三步，尝试、尝试、再尝试。

8.2.7 Ballpacking «»

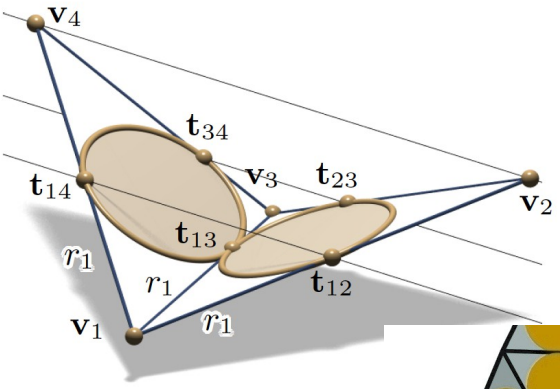
这是一个完全不同的优化参数: Ballpacking, 这个参数主要是针对三角网格，当然也可以用于其他类型的网格。这里我会以三角网格为范例来讲解。

首先我需要解释什么是内切圆，一个三角形的内切圆是三角形内最大的与三边都相切的圆。[如下 图.13]



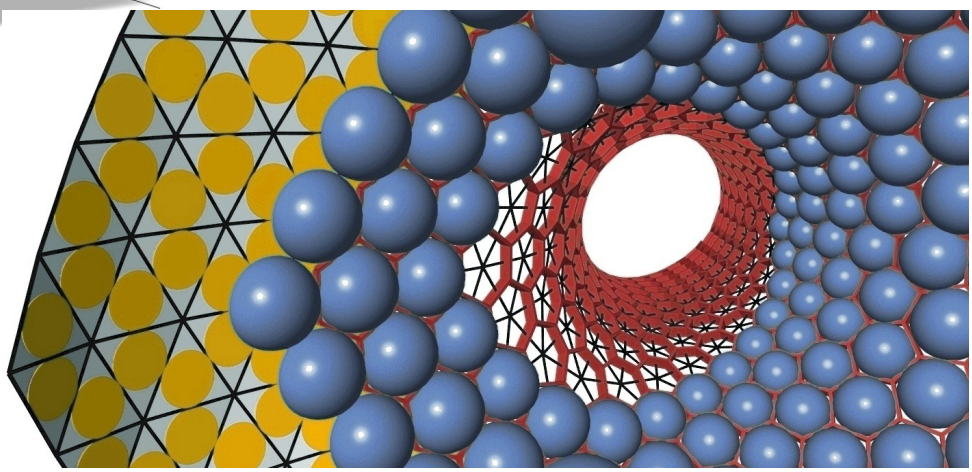
图片.*13* - 内切圆图。**左图**: 内切圆圆心放置与三角平分线的交点。**右图**：垂点与边缘切点的距离是相同的，边线 *A&D* 为内切圆的半径，所以 *A=D*，边线 *C* 为公共边且平方顶点角，所以 $\vartheta_1 = \vartheta_2$ 且 *B = E*。

当执行 Ballpacking 约束优化时，使得相邻的三角形内切圆接触且相切。如上图所示，顶点距离内切圆切点的距离是相同的，当三角形变成球体充填的一部分时，所有相交三角面的相交顶点至切点的距离也是相同的。因此在下图： $\mathbf{r}_1 = ||V_1 - t_{14}|| = ||V_1 - t_{13}|| = ||V_1 - t_{12}||$ $\mathbf{r}_1$ 视为圆心位于顶点的圆半径，我们最终的球体充填结果：相邻的球 $\mathbf{S}_i,\mathbf{S}_j$ 相切与 $\mathbf{t}_{ij}$， T 这样我们称之为一个球体充填[如下图.15 所示]。最终视觉上平衡网格，另外促成另外一个重要的建筑熟悉，即为自由扭转节点。



图片.*14* - 球体充填内的相邻内切圆。

图片.*15* - 球提充填内切圆（黄色）范例，由圆心（红色）连接而来的最终球体（蓝色）与六边形网格晶格。



如果我们实际来施工一个网架，让梁沿着边线放置，所有梁靠近节点的相交点是一个重要的问题，我们期望所有梁的对称平面都很好的汇交于公共的节点，这个加点的垂直轴刚好通过一个顶点，把这种梁称之为

带有自由扭转节点的支持结构。

如果我们使用 Dual 细分把一个规则的球体充填网格转换为一个六边形网格，这就非常容易实现，且我们会在后面的章节 8.5.2 中举例说明，在新的网架中所有的轴线都是前面内切圆的轴线，而且相邻内切圆轴线都都共面、正交于内切圆功能切线。因此公共平面所对应的沿着边线的梁的对称平面上的顶点与节点都是自由扭转节点。在 Evolute 主页我们有提供更多的介绍， "Packing Circles and Spheres on Surfaces"。

技巧：你可以使用 *_etBallPackingMeshExtractBalls 指令提取所有的充填圆球，完成之后你可以使用 Sellast 与 SelDot 指令找到球体。*

额外技巧：作为 Dual 细分的补充，球体充填也是一个非常好的六边形细分工具。

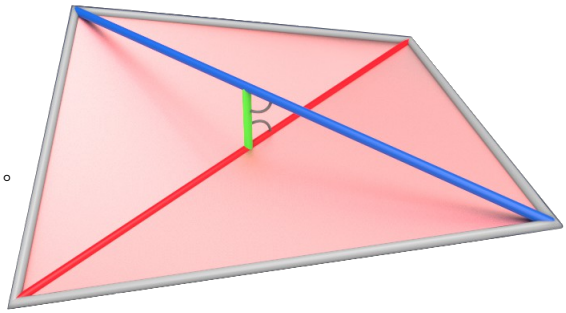
专业技巧：在 EvoluteTools 程序目录你能找到一个名为 ScriptingExamples 的文件夹，你能在里面找到一个名为 Incircles 的 RhinoScript 文件，选择一个网格运行这个脚本，将会在一个新的图层提取所有的内切圆。如果你想了解更多的 EvoluteTools 脚本应用方法，请浏览有 Marko 编写的 EvoluteTools Scripting Primer!

8.2.8 Coplanarity

这是一个用于局部共面优化的约束参数，了解详情请浏览 8.4.7-SetVerticesCoplanar章节。

8.2.9 Planarity

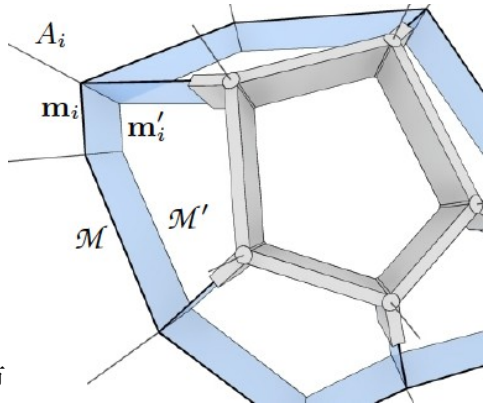
在第 8.1 – 优化简要介绍章中我们有讨论过这个约束参数，这里再次总结一下。这个参数用来优化网格块面的平直度，平直度是测量一个块面两对角线之间最短的距离。对于大部分项目来说平直度是一个非常重要的约束参数，直接影响面板造价。



8.2.10 Conical 与 Circular Optimizations «»

当给 Conical 参数设置一个强度值是，优化后的网格会变成一个圆锥网格，当提升 Circular 参数时网格会变优化成圆形网格。好吧，继续!

我知道你很难在第一次接触圆锥网格与圆形网格你可能会一头雾水，对与建筑施工而言，意义非凡，我相信你值的在此多花一些时间来了解。



主要目的是为了得到一个能够整洁平行的网格偏移，对建筑而言这是个非常重要的参数，因为建筑都是多层的。网格只是一个简单抽象的数学概念，就实际施工而言充其量只能做一个单层，为了让抽象的网格满足实际施工设计，偏移是必不可少的。

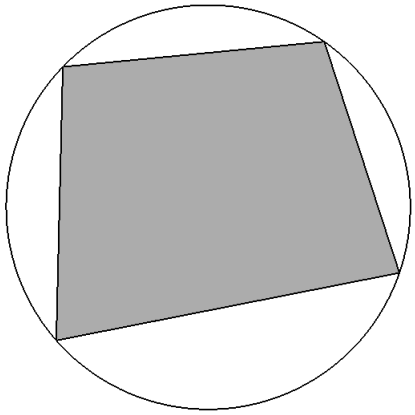
在网架实现中，接近平直的面板（成本较低）由沿着网架边线排布的梁（一般都是标准的横截面）支撑，梁都是围绕一个中心平面对称布置。中心平面由网架边线相交的理想节点轴线[例如自由扭转节

点-详情查看 8.2.7-Ballpacking «»]，节点轴线是连接网格对应顶点与偏移点的直线。[如右图所示]

可以很明确的定义顺滑曲面的偏移：以曲面上每点法线方向等距的移动一个曲面，但网格偏移并不是这样直接移动的，网格偏移有几种不同的定义且属性都不一样，与 EvoluteTools 相关的有两种：

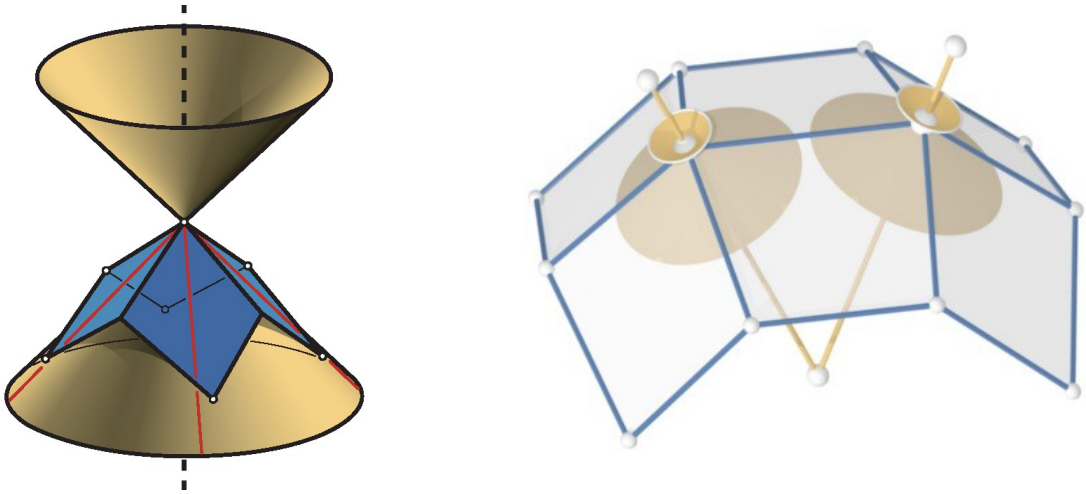
- **顶点偏移**：偏移网格与网格的顶点之间有一个相同的间距，这种属性的网格叫做“圆型”
- **面偏移**：偏移网格与网格的面之前有一个相同的间距，这种属性的网格层之为“圆锥形，”圆锥形网格带有自由扭转节点！

并不是任意一个平行网格偏移都有这样的属性，这是两个特别的参数，因为你可以使用 EvoluteTools 优化你的网格让他具有这样的属性。这里我不会讨论其他类型的网格偏移，边线偏移等，因为你可以在这里 this fascinating paper 看到更多由 Helmtt Pottman 等编写的内容。



不在此讨论的真正原因是圆形网格具有顶点偏移的属性（你可以在 Architectural Geometry 中找到图文并茂的描述）。他可以帮你了解优化的基本属性，果每一个四边形网格都拥有一个环形圆那么这个四边网格就可以叫做一个圆形。[如左图所示]

当所有的块面都相切于一个顶点，且这个顶点相切于由一个通过顶点轴的环绕旋转而成的圆锥体时圆锥网格叫做圆锥柄[如图片 16a 所示]，如你所见下图 16b,在圆锥网格的两个连接顶点的轴放置在一个公共平面上，因此，任何通过这个边线的梁都会有一个与这些轴线重叠的对称平面，这些轴与从网格偏移衍生的结构都具有自由扭转节点。圆锥网格非常的光滑!



图片 ***16*** *- a)* **左图**：*4* 块面相切与一个通过他们公共顶点的旋转圆锥体。 *b)***右图**：在一个圆锥网格的两个相邻顶点的轴都放置在公共平面。

8.2.11 Normal Closeness «»

这个优化参数用于最小化顶点从其当前顶点法线的背离，或者说，它尽力限制顶点在其法线方向的偏移，这并不是一个必须的全局优化约束。你必须选选择你想要利用 etSetVertexRestrictedToNormal 指令或  来限制约束的顶点，如果你想要这个参数影响到整个网格，你仅需要选择全部的顶点。

对于控制某型顶点的定量位移很有用，有时候使用使用 etSetVertexFixed 固定一大群顶点太约束，这样不会应许他们任何的位移，利用 Normal Closeness 参数你可以运行优化程序内外的条节点而不会让点移动穿过曲面。

这个功能仅限 EvoluteTools Pro 版有效。

8.2.12 Ideal Edge Length «»

类似 Normal Closeess 参数，Ideal Edge Length 不会影响整个网格。

这个优化选项仅对你使用 etSetEdgeLengthOptimization 指令  挑选的网格边线其作用，你可以选择你期望等长的边线。

等长边线长度优化在 etOptionsTooggles [查看章节 8.3.9]设置中有一个相同的名字，两者选一，如果在 Toggle 中没有设置值，这个算法会让所选择的线尽可能的等长。如果你没有预设长度需求后者更加有用，因为优化程序会为你选择的网格与目标几何找到一个最合适的长度。

这个功能仅限 EvoluteTools Pro 版有效。

8.2.13 Ideal Panel «»

对于三角面与四边面的优化这是一个非常“精巧”的参数，优化三角网格能得到等边三角形，且边长接近 etOptionsToggles [查看章节 8.3.9]中的 IdealEdgeLength 参数设置。四边网格优化能得到尽可能等长的边长。

我使用“精巧”词是因为具有等长尺寸的完美面板的曲面面板仅在特定的条件下才会产生，如果设置相对其他参数太高的权值可能会造成与预期相反的结果。

这个功能仅限 EvoluteTools Pro 版有效。

8.3 优化切换

不是修改 Options Importance 设置中的相关优化参数，而是修改 Optimization Toggles 对话中优化程序自身特征。你可以使用 etOptionsToggles  进入这些设置选项，可以使用 etOptionsReset  来恢复默认设置。

8.3.1 Fairing

这个选项的两个切换设置会影响 FairnessCurvature 与 FairnessCurvatureVariation 的计算，默认设置为

absolute,这个模式整顺参数功能如前面一章所述，FairnessCurvature 最小化曲率（例如网格多重线矫直）与 FainessCurvatureVariation 最小化曲率变化。

另外一个选项为 **relative** ,这个选项选择当前所使用的网格状体作为整顺参数的参考值。或者说，当这个选项开启时 FairnessCurvature 会以当前网格的曲率来最小化不同的曲率差异，而不是将曲率差异最小化接近 0。同样 FairnessCurvatureVariation 也是一样，这个参数将最小化不同的曲率变化率至当前网格的曲率变化率。

如果你有想要得到一个和当前网格一样的平顺与曲率的网格，但仍然有一些参数需要优化，例如平直度，这种情况使用这个参数非常有用。这样比使用全局 Original Closeness [查看章节 8.2.3]约束更加能保证网格的主要形态且能给优化器更多的自由。

警示：当使用 RelativeFairing 切换选项时所选择的参考网格是你在运行 etOptmize 指令时选择的网格，也就意味着你每运行 etOptimize 一次参考曲面就会重设一次。重要的是它增加的迭代[查看下面的第 *8.3.5* 章]次数会让优化运程序持续运行至收敛，否则最终的效果可能会持续偏移，会与你想要的结果偏离。

8.3.2 FairingMeasureScaleInvariant «»

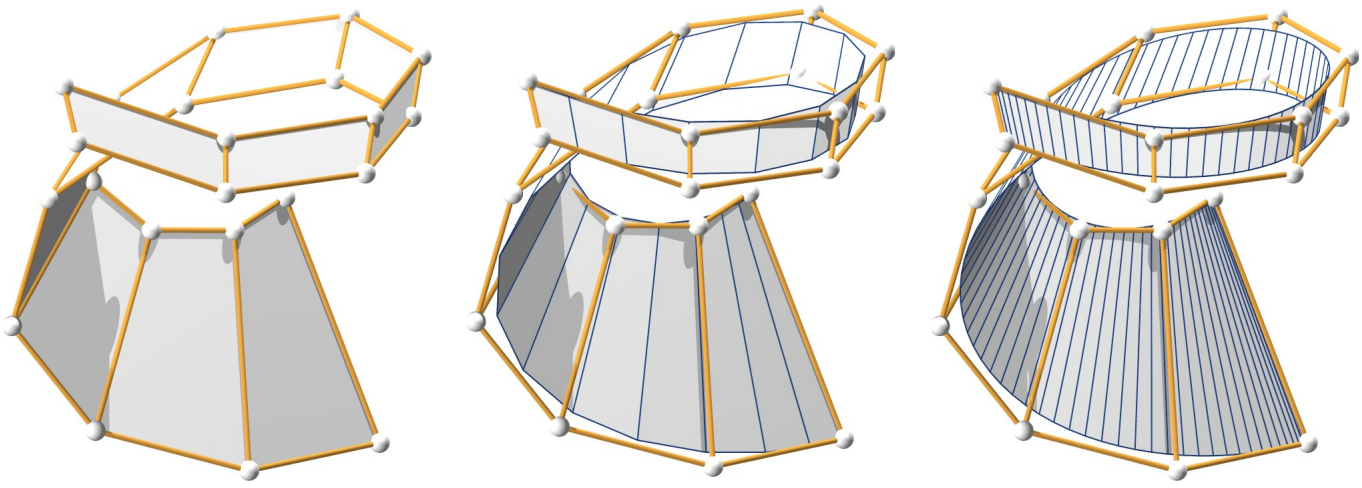
和 Fairing 切换一样，这个设置会影响 FairnessCurvature 优化参数，一方面会使整顺参数尽可能的集中高曲率区域的顶点与边线。当这个设置为 ON 时，影响稍稍会减少。

默认的设置值为 **off**,优化程序会使用一个变化比例测量，这样在曲率越高的地方边线越短。

当设置为 **ON** 会使用一个固定的比例曲率测量，这样会得到相同长度边线的分布。

8.3.3 FairingMeasurePerEdgeScaling «»

这个设置用于带状网格的平顺且通常可能会切换。带状是增加了比较多的单排细分而得到，总的来说会维持一个较小的平均长度。当单排细分不断增加时带状网格多重线也会变的越来越短，相对于带状边线比例会越来越大。[如下图所示]



如果使用整顺参数让每条边线都变得接近相同长度，这样会沿着带状方向给出越来越多的权值，因为增加它们长度的差异。

当设置为 **off** 时，使用主要的整顺参数会让每条边线都会趋向等长。如果设置为 **on**，优化程序会对每条边线都使用单独的整顺缩放比例。这会在多个指令中用到，在 _etSubdivide 中的 Strip 细分规则。

这个功能仅限 EvoluteTools Pro 版。

8.3.4 PlanarityMeasureScaleInvariant «»

这个切换选项用于 Planarity 优化参数，默认的设置为 **off**,这个设置会让优化程序最短化网格面对角线之间的最短距离。如果设置为 **on**,优化程序会最小化网格面对角线最短距离除以网格面对角线长度的值，或者说当网格面的尺寸增加其平直度公差背离值也会增加。这反映了真实的物理属性，越长的材料允许的变形也就越大。

这个功能仅限 EvoluteTools Pro 版。

8.3.5 Iterations

这个选项用来定义执行 etOptimize 指令是运行的迭代的次数，通常你会设置比较低以便于你可以估计设置对网格的影响，必要时能及时调整。这个参数越低，调整过程可控性越好。然而，正如前面所提到的，如果你有使用任何的优化参数或把当前网格状物作为一个附加参考，你会想到增加倍增值，因为每次执行优化后当前网格状态都会被重置。

注意：增加这个值会造成 etOptimize 运算时间加长，但整体的优化时间会降低。

8.3.6 DefaultBoundaryCloseness

用于切换 CurveCloseness 优化参数的效果，默认设置为 **on** 且大部分场合都建议为 On，当优化是所有网格边缘顶点都会自动的标记为曲线点 [查看章节 8.4.1 - SetVertexCurvePoint]。这会让边缘顶点捕捉到距离最近的参考曲线 [查看章节 8.2.2 - Curve Closeness]，进而有优化程序会自动的探测且将所有的曲面边缘作为参考曲线，这会是一个乏味的工作但也能减少一些控制。

当设置为 **off** 边缘顶点不会自动标记且你必须使用 etSetVertexCurvePoint 来定义那些顶点需要捕捉到距离他们最近的参考曲线。另外参考曲面边缘不会自动的增加为参考曲线，这个设置允许更加需要定义边缘，当作业于用球体充填网格时非常有用。

8.3.7 CutOffSurfaceDistance «»

这个设置用来定义 SurfaceCloseness 优化参数的约束距离，Closeness 将仅优化与参考曲面距离值在这个范围之内的顶点，超过这个范围见不会被 Closeness 优化选项计算，但依然会被他的优化选项计算。

默认设置值为 1e+154 或 1 x 10^154，这是一个非常大的数值除非特殊情况，所哟的顶点都会在这个范围。通常你不需要修改这个值，如果你想在多个参考曲面见混接一个网格或是用来做找形分析的时候你会想到降低这个值。

8.3.8 CutOffCurveDistance «»

这个选项与 CutOffSurfaceDistance 功能接近，这个选项用于定义 CurveCloseness 参数，用来定义距离执行曲线逼近优化的顶点标记为边缘的距离范围，所有超过这个距离的顶点不会被纳入 CurveCloseness 优化计算。

和前面的切换值一样，默认值是一个非常大的范围且满足绝大部分的场合需要。如果你想要设置一个特定的范围来让顶点捕捉参考边缘时，你可以根据需要设置这个值。

这个功能仅限 EvoluteTools Pro 版。

8.3.9 IdealEdgeLength

这个参数的设置值与优化参数的同名选项联合起来用，当这个值设置大于 0 且 IdealEdgeLength 优化参数[浏览章节 8.2.12] 会给定了一些强度值，所选的辨析会将长度优化接近这个设置值。

如果保持这个参数设置为 0，优化程序会检查所选目标边线的长度且给出一个最合适的长度值。如果你没有特定的面板尺寸要求，我会推荐你设置这么做。例如，如果你仅在网格细分前期尝试均等边线长度，这样的设置会很完美。

提示：仅仅对使用 etSetEdgeLengthOptimization 指令  所选择的边线起作用。
[浏览下面的 8.4.6 章节]

这个功能仅限 EvoluteTools Pro 版。

8.3.10 StripFairingScalesRatio «»

这个设置与 Strip 细分[浏览 7.2.9]规则联合起来使用，如果 FairingMeasurePerEdgeScaling 开启，这个条形网格（沿着条形长边方向放）整顺参数使用 Strip 细分规则以这个值为比例缩小网格。如果这个参数值设置为 0，会使用 条形边长边与短边方向长度的平均值自动计算一个缩放比例。

8.4 局部约束

局部约束与全局约束想法，仅仅对你所选择的到网格其作用。一般来说，用来调整一个网格，给用户较高的自由度来控制网格的修改。

主要窍门：下面的所有指令都可以接受顶点作为输入（例如在任何指令开始前使用 etSetVertex）也可以预选顶点，也就是说你可以使用任何的 Rhino 或 EvoluteTools 的选择方式（例如 etSelectMeshBoundary）来获取你想要的顶点然后执行局部约束指令。二选一，如果你没有选择顶点，执行指令时会提示你选择顶点，但你只能使用鼠标的方式来选择。

主要窍门#2:如果你想要取消预标记的顶点，再次运行指令然后安装 CTRL 选择你要取消的顶点（和 Rhino 的选择/取消选项模式一样）。

8.4.1 SetVertexCurvePoint

这个指令 etSetVertexCurvePoint  将网格顶点指定为优化的曲线点。如果有设置 CurveCloseness 优化参数值，在优化期间，用这个指令所标记的顶点与最接近参考曲线的间距会越来越小。也就是说，所指定的顶点价格会朝最近的边缘线移动。如果 DefalutBoundaryCloseness 开关为 on ,所有的边缘顶点 会自动的被指定为曲线点，且避免你收到选择的麻烦。

另外，如果你有 EvoluteTools PRO,可以利用 etSetReference 沿着取消对齐网格内部顶点，如果你想对齐用于建筑重新定义栅格结构的面板布局时非常有用。

8.4.2 SetVertexFixed

我有讲过一些办法用来约束网格顶点的偏移量，但如果你想完全固定一些顶点你需要使用 etSetVertexFixed  ，这个指令允许你为优化程序指定用来固定的网格顶点，使用这个指令标记的固定的网格顶点，在所有的 etOptimize 过程中都不会被改变。

8.4.3 SetVertexCorner

这个指令 etSetVertexCorner  为优化程序将指定的顶点作为网格的转角，优化期间，以指定的边缘网格顶点会从所有整顺参数项中排出，会阻止优化程序矫直所有围绕他们的这些网格边缘顶点，从而创建一个转角。如果想要在优化过程中精准的保留转角的坐标值，优化时需要配合 etSetVertexFixed 指令。

这个指令有两个选项：

AutoDetect:设置为 ture，EvoluteTools 会更所设置的 KinkAngle 沿着边缘边线查找符合设定条件的顶点。

AutoDetectKinkAngle:如果开启 AutoDetect，所有边缘上的尖角大于这个选项的 KinkAngle 设置角度的顶点都会被选中。

8.4.4 SetVertexRestrictedToNormal «»

_etSetVertexRestrictedToNormal'  用来限制顶点在其法线方向位移。执行这个指令时会高亮预选的顶点，使用这个指令选择的顶点会约束其在优化过程中沿着逼近的顶点法线方向位移，约束强度值由优化参数设置 NormalCloseness 确定。如果你有一个网格已经优化至你所期望的形态，但平直度还不完全符合要求，你可以设置 NormalCloseness 代替整顺约束且联合利用平直约束将改进面板的平直度而不会明显的改变网格的形态。

这个功能仅限 EvoluteTools Pro 版。

8.4.5 SetEdgeFeature «»

到现在为止所介绍的大部分功能都用于顺滑自由造型设计，但如果你的目标几何有一些变化锐边，或有一些折角，这时候你就需要着手设置边线特征指令。

你可以是使用指令 etSetEdgeFeature;  来指定网格边线作为折边。使用这个指令你可以标记网格边缘在优化过程中维持折边，会阻止任何整顺参数对这些已标记的边线作业。也就是说，在顺滑的网格内会包含一个折边特征。当执行时，会高亮已经标记过的特征边线，你可以使用多个选项来选择或取消选择。

SelectByVertices: 这个切换允许同时选择一个顶点来以选择一条多重网格边线。

SingleEdge 允许你选择一个用来标记的单一边线。

Polyline 一次选择以用来标记的多重边线。

ClearAll 取消所有标记的边线。

AddAll 标记所有边线。

Autodetect 角度提示，用于自动标记两个临近相交其尖角大于指定值的所有边线，其他的边线不会被标记。

简单的关闭整顺穿过折边并不表示这些各自的顶点会自动的捕捉到你目前的折边边线，有必要定义顶点作为附加曲线点的且定义需要定义折边 ianx 作为附件参考取消。

这个功能仅限 EvoluteTools Pro 版。

8.4.6 SetEdgeLengthOptimization

你要使用指令 etSetEdgeLengthOptimization  来设置你想要等长[浏览 8.2.12]优化的网格边线长度，当你运行这个指令时，会高亮已经标记过的边线，可以使用以下的选择与取消选择工具。

SelectByVertices: 这个切换允许同时选择一个顶点来以选择一条多重网格边线。

SingleEdge 允许你选择一个用来标记的单一边线。

Polyline 一次选择以用来标记的多重边线。

ClearAll 取消所有标记的边线。

AddAll 标记所有边线。

Invert 对当前的选择反选，已经标记过的边线会被取消。

这个功能仅限 EvoluteTools Pro 版。

8.4.7 SetVerticesCoplanar

对用控制你网格中顶点的移动 *'_etSetVerticesCoplanar'*  是一个非常有用的工具，对建筑来说非常有用，因为他可以对齐一群对点于例如楼板的平板构件。总之，这个工具允许你标记一组你想要对齐与指定平面的顶点。

运行这个指令时会显示已经标记过的作为对齐平面的顶点，如果有的话，允许你浏览这些已标记的顶点。如果是第一次标记，会提示你至少要选择两个顶点来创建一个新的设置。这个指令包括以下选项：

Add 允许你增加其他的顶点至平面共面顶点。

Remove 从已定义列表中清除当前显示顶点群。

Previous 让你切换至前面一个已设置顶点群。

Next 切换至下一个组已设置顶点。

Planetype 让你指定共面约束：

GeneralPlane 不会增加任何的约束，这个选项会判测最合适的平面。

ParallelToYZPlane、**ParallelToXZPlane** 与 **ParallelToXYPlane**, 优化顶点平行于至各自的指定坐标平面。

NormalToYZPlane、**ParallelToXZPlane** 与 **ParallelToXYPlane**, 优化顶点垂直于至各自的指定坐标平面。

FixedPlane 构造这个约束平面会提供以下选项：

SelectPlane 会询问你选择一个平面曲面且将固定这个面为平面曲面。

SelectPlaneNormal 会询问你选择一个平面曲面且将固定这个面的法线至选择平面曲面的法线（将允许移动这个面）。

HorizontalWithFixedZCoordinate 将约束这个面至水平，且会要你为这个面设置一个高度，对于 SeletPlane 使用一个水平面是最快的方法。

Importance 让你为当前共面约束设置一个优化强度值，这个强度值与 Coplanarity 约束值的乘积来决定共面优化强度。

ShowPlane 增加一个平面曲面至当前文件，显示当前所使用的优化平面。

ShowDistancesFromPlane 从当前平面显示一个文本数字，用来显示当前约束顶点距离当前约束平面的实际距离。

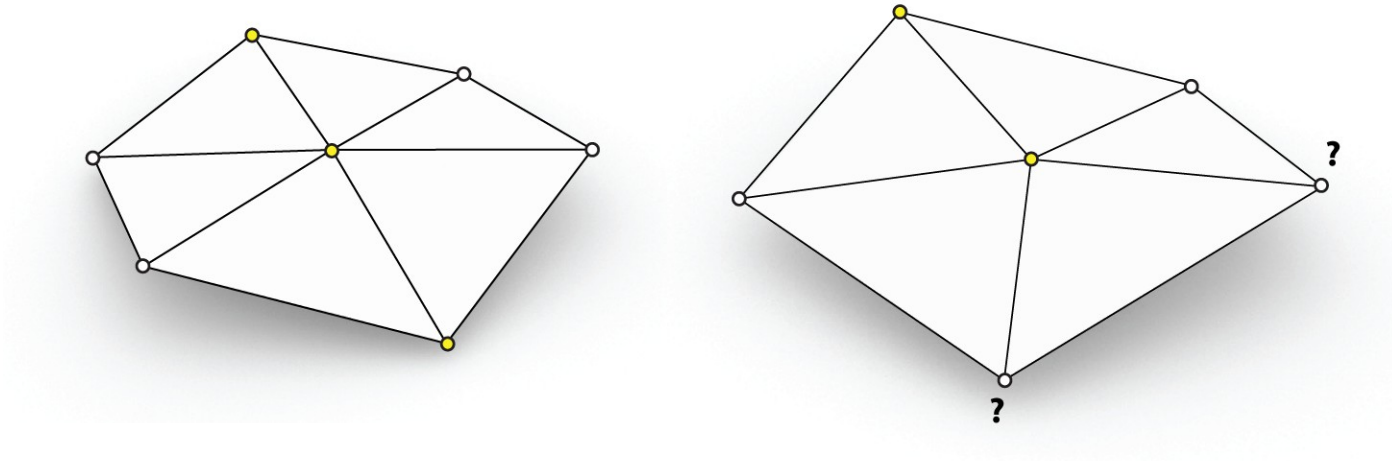
一般情况下，修改网格连通性期间共面约束顶点列表会被保留，如果一个共面约束顶点列表包含的顶点少于所需要的顶点数（根据所在面定义决定）在修改网格连通性后会被移除。

这个功能仅限 EvoluteTools Pro 版。

8.4.8 SetVerticesFairing «»

指令 *etSetVerticesFairing*  允许你选择网格顶点用于附加顺的滑优化（最小化网格中的扭结），最主要的用途是能对你的网格顺滑提供一个非常细腻的控制，很多情况下非常实用。

为了更好掌握如何使用这个工具，考虑如何穿过 3 个顶点组以及由其所定义的角度（例如扭结）来计算顺滑是很有帮助的。也考虑位于数条边线所相交的单一中心顶点，在相交的顶点会有一个偶数的维度（例如偶数的引入边线），顺滑优化检测从边线另一尾端的顶点组所构造成的中心顶点。



然而，对于奇数维度的顶点，优化程序无法定义对立组数且在顺滑算法中会被忽略[如上图所示]，当然你可以使用 etSetVerticesFairing 来手动选择顶点，另外，在边缘的顶点仅其他边缘顶点是成组顺滑。如果你想顺滑一个内部网格多重线至一个边缘边线，你需要使用这个指令。

它还可以用于沿着“虚构”网格多重线顶点顺滑，我知道，知道，请稍等一下，你可以选择一些并没有由实际网格边线相连的顶点，顺滑优化会把他们当作一个多重线来优化且会移除这些扭结。

注意：选取顶点的顺序很重要，当你选取顶点时想象自己在画这条你想要顺滑的多重线。

当唤起时，这个指令会显示已经定义过的顶点列表，如果没有的话，或提示你至少选择三个顶点来创建一个新的列表，你会用到下面的选项：

Add 允许你增加其他的顶点列表值附加顺滑优化。

Remove 清除当前显示顶点列表。

Previous 让你切换至前面一个已设置顶点列表。

Next 切换至下一个组顶点列表。

Importance 让你为当前顶点列表设置的优化强度值，这个强度值 与 FairnessCurvature 的强度值相乘觉得最终的优化参数强度值。

一般情况下，修改网格连通性期间共面约束顶点列表会被保留，如果一个顺滑约束顶点列表少于 3 个顶点，

在修改网格连通性后会被移除。

这个功能仅限 EvoluteTools Pro 版。

8.5 优化范例

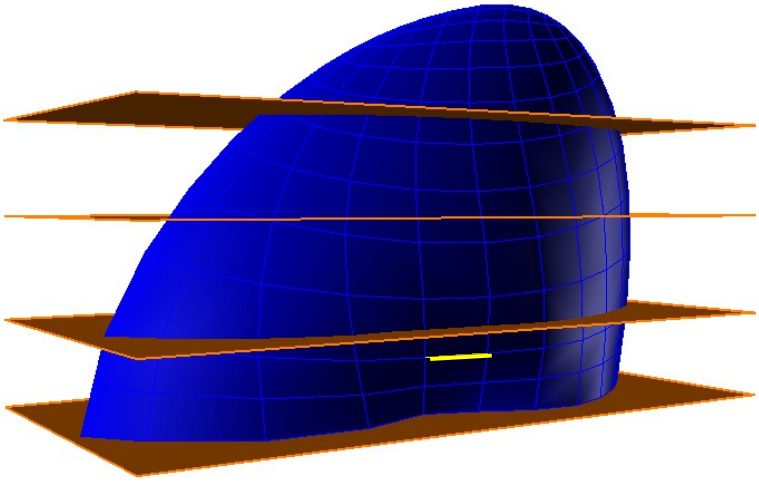
闲话少说，范例如下：

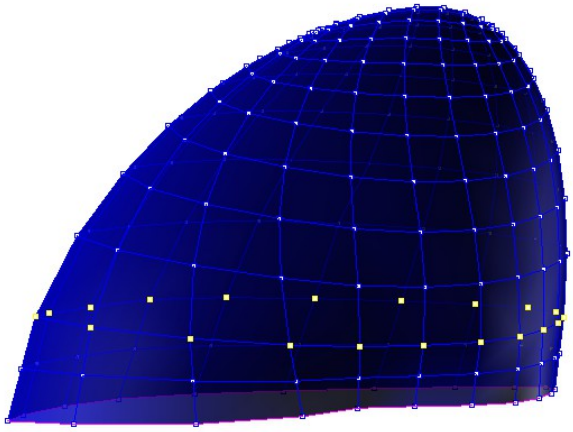
8.5.1 范例 9 - Planarity、Coplanarity、Fairness

这个范例我们会从一个最初的建筑表皮网格开始，容然后优化其平直度，同时会利用建筑楼板层对齐面板布局。

1. 打开文件名为 Example9 的文件，这个文件包含一个参考曲面、最初的网格与用来对齐面板的一组平面。

2. 首先使用 etSetVerticesCoplanar 指令  来设置共面的顶点组，但不是一次选择所有要处理的顶点，你要使用 etSelectMeshPolyline  工具来预选每一组，选择这个工具，然后在网格多重线选择一个接近第一层楼板的水平边线。[如下图所示]





3. 以 polyline 的方式单独选择所有的顶点，非常方便，现在你需要运行共面工具  ， 指令行会提示你“Planetype”与 “importance”值，强度值设置为 1， 但你需要改变平面类型，选择“Planetype”选项。

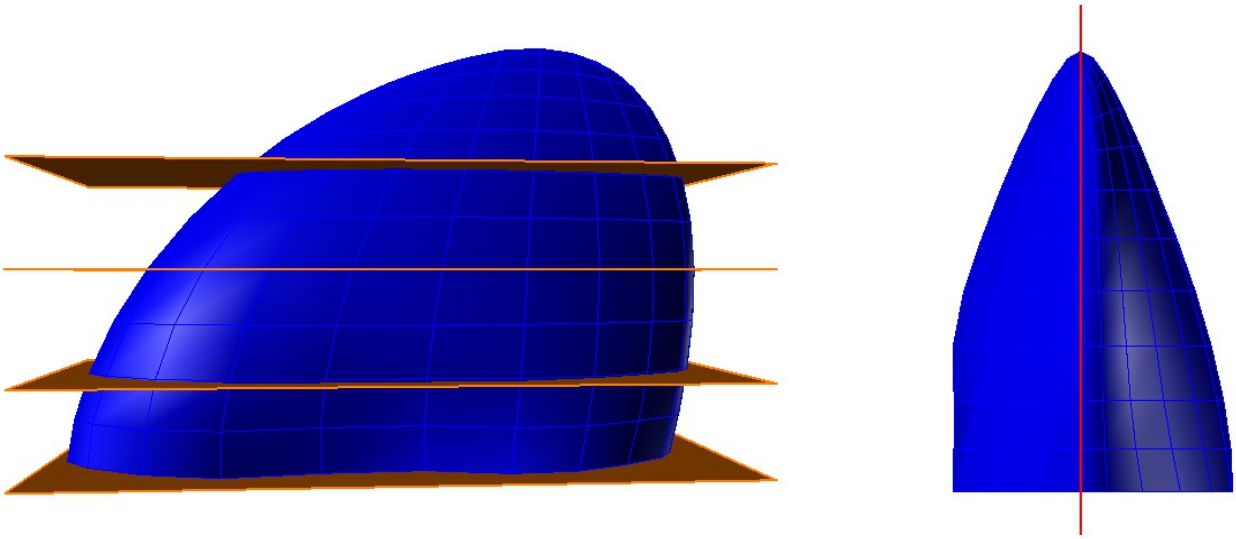
4. 现在指令行提示栏会给你一个平面类型列表，章节 8.4.7 中详细介绍。这里选择 “Fixdplane”然后按 [Enter]。

5. 接下来会给你 3 个选项来的定义 FixdPlane,因为你已经画好了平面，所以这里简单的选择 "SelectPlane"。

6. 选择第一层楼板平面然后查看指令行的提示：“Preselected grips set to become coplanar.”

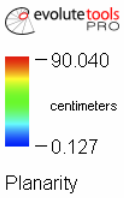
7. 现在为其他的楼板与中心对称面重复相同的动作。

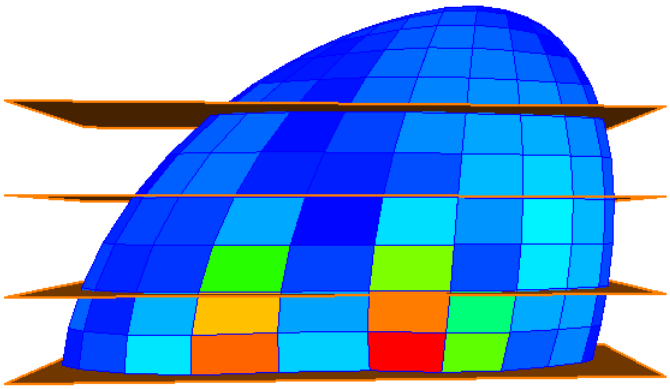
8. 请确定你的优化设置有恢复默认  ，然后设置 FairnessCurvature 为 0.2, FairnessCurveVariation 为 0.4 与 Coplanarity 为 0.6，提示：如果这里单独设置整顺参数会造成网格多重线变得既直且扭结或太曲。这个组合设置能平衡这种问题。



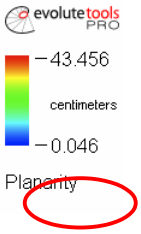
9. 多次运行优化指令后，你的网格会很好的对齐与中心与不同的楼板。

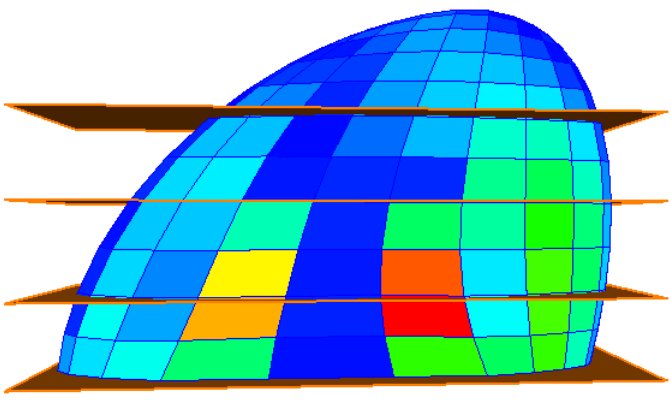
10. 网格的排版现在看起来比较好了，当你要记得我们希望面板变得更加的直，让我们使用 etAnalyzePlanarity 指令  来检测。





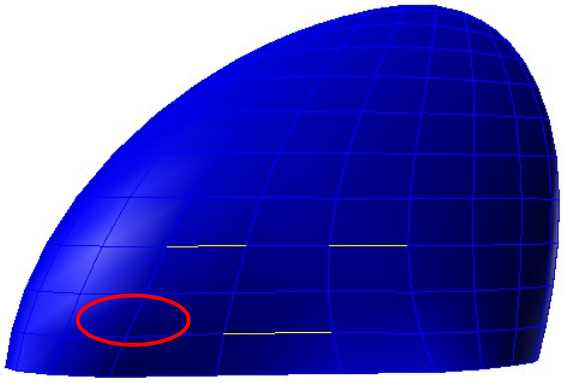
11. 这里有一些问题教明显的区域，线将 Planarity 优化参数设置为 1，再次允许优化（确定有在网格分析对话框中点击了 Min-Max Rangelai 重置显示比例）。



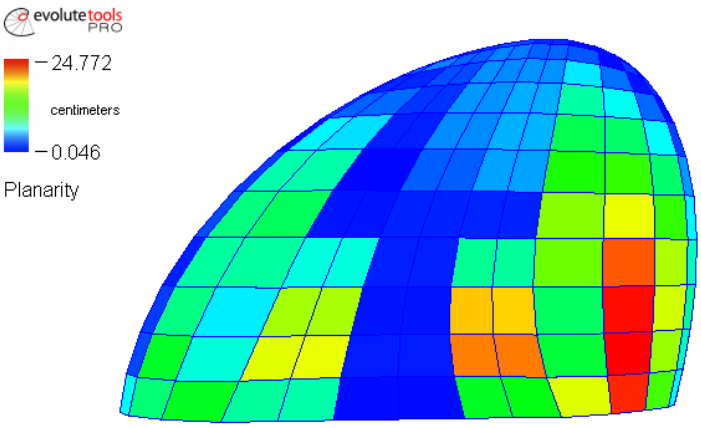


12. 你会看到有效果更好，但 planarity 的最大背离值还是太高，但已经有减半了。实际上对于逼近目标几何面板的平

直度还是太大。



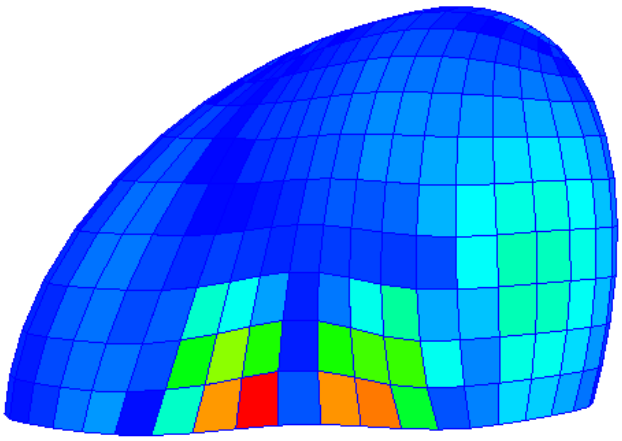
13. 为处理这个问题，使用 Mesh Loop Cut[章节 7.3] 选择如右图所示的 3 条边线进行网格分割，这一步不会无需额外的优化参数设置即可改善平直度的问题。



14. 再次优化之前，你需要切换 FairingMeasureScaleInvariant «» 设置为 on ,这个造型沿着脊线区域比较相同相对平坦的一边已经有比较高的曲率度，如果不切换这个设置，优化程序将会汇集这些较小的边线，这样将会从问题区域移动我们刚刚增加的细节太远。

15. 更多优化。

16. 多次重复执行 13&15 步，你将要注意两个事情。第一，这里仍然有一些有问题的面板，但大多数的网格都已经变得更加平直了。第二，这里仍然有一个突起的行政，位于你前面所设置的共面的多重直线内！



这是因为你在整圈分割时增加的新的顶点，但没有纳入共面组内，你手动添加。

17. 选择网格，然后执行共面工具  ，会显示你之前设置的第一组顶点且你会看到新的位凸起位置的顶点没有被选择，点击 Add，然后选择新的顶点，然后点击回车，接着选择适当的平面。
提示：这里不要将其放置到其他组，当你可以把它们设置到相同的共面平面且会有相同的效果。

18. 其他凸起区域的新顶点做相同的操作，然后执行优化，将会整顺这个凸起。

这个网格已经整理的很顺但还不足以变成一个足以能建造的成品，但就如何使用优化工具方面希望能给你一些启示，如果你想继续深入，建议你在水平方向进行一些和垂直方向相同的整圈分割与操作，这样会改善平直度。

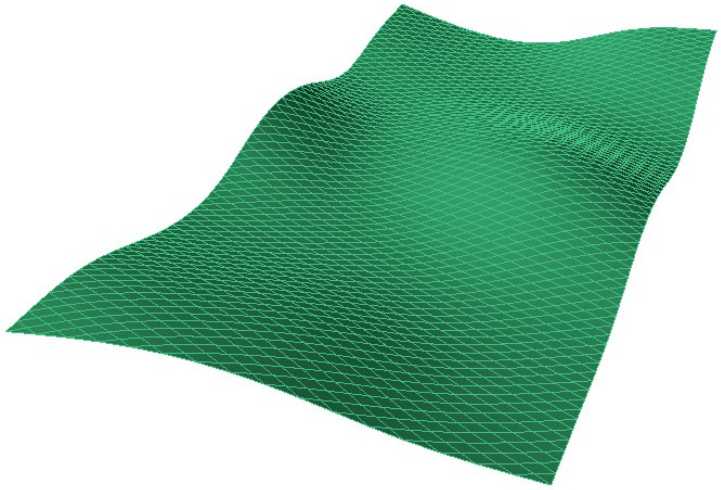
8.5.2 Hexploration – Ballpacking & Ideal Edge Length

如果你习惯视频教程学习，我不会太推荐 Evolute 在 YouTube 的视频教程。这个范例将会讲述不同的侧重点：我们将更多的关注如何在没有严格的参考边界的情况下得到一个边线长度相等的完美的球体充填网格。不管多么精彩，下面开始我们的范例操作。

1. 打开范例文件 PrimerExample_10，这个文件包含一个标准的“飞毯”参考曲面且有一个四边网格面等待被优化。这有转角顶点需要标记且有一些优化参数设置已经为你准备好了。

2. 打开优化设置仔细查看，连 FairnessSpring 的破折号你都需要留意，这个范例，不会有其他的顺滑参数会造成网格多重线的小偏移，不相信吗？试试你就知道了。

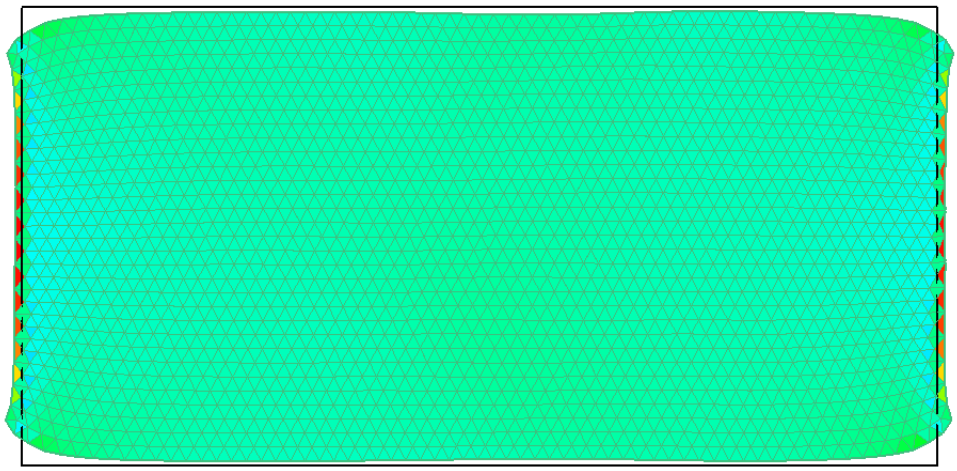
3. 优化几次直到收敛，然后使用细分工具中的 Diagonalize 算法，你的网格细分后看起来像这样：



4. 为了得到一个六边形的网格你线需要一个三角形网格，因此你需要使用这个工具：  *'_etMeshAddDiagonalLinesParallel'* 且 interval=0, 在曲面的长方形选择连个顶点，你的网格会变成一个三角形网格。

5. 现在看起来有点失控，但并没有跑太远，设置 CurveCloseness 为 0.5，设置 FairnessSpring 为 0， Ballpacking 为 1 且 IdealEdgeLength 设置为 1，取消转角顶点与固定。

6. 为了让 IdealEdgeLength 参数有一些效果，我们必须标记你想要等长优化的边线，这个范例中选择所有的边线！！就这样做！点击  或运行 etSetEdgeLengthOptimiztion 指令，指令行对话框点击：'SelectionMethod = SingleEdge'修改选择你期望等长的边线。下一步选择'AddAll' 因为你想要所有的边线等长，然后点击 [Enter]



7. 想要了解 IdealEdgeLength 参数的优化情况，打开网

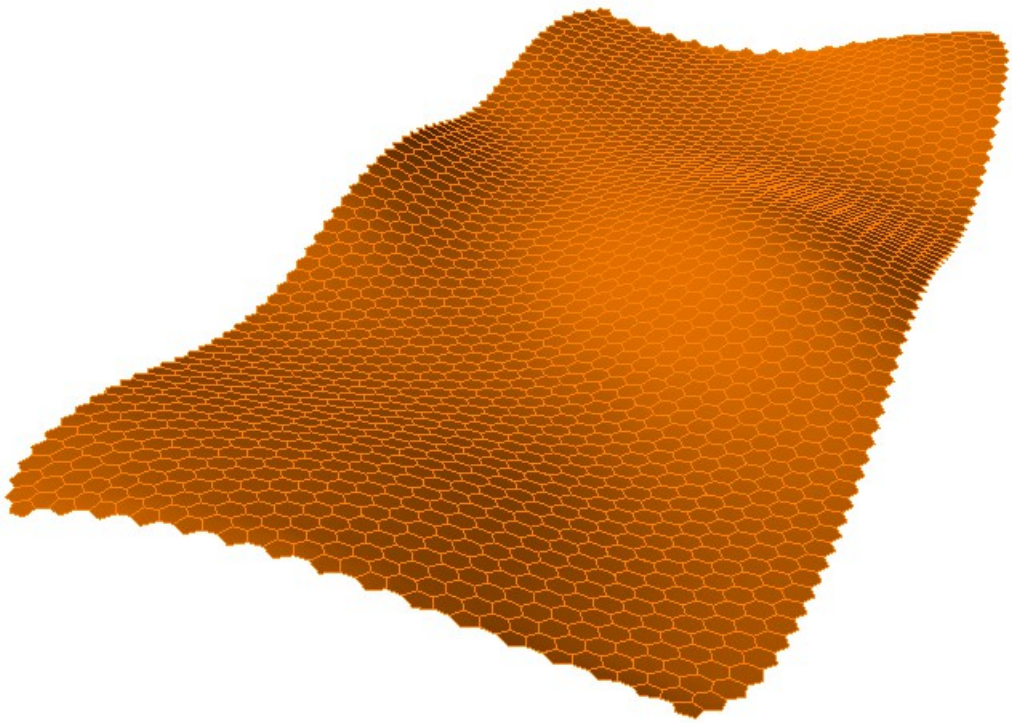
格分析窗口（选择  ）然后在选择“Mode”栏点选“Edgelengths”，这个分析图每一块面都会根据其最长的边线显示颜色，你的网格现在看起来应该如下图所示，目标曲面边缘显示为参照。

8. 你可以看到颜色的分布，除了两侧之外其边线的颜色都非常统一，网格也从边缘沿着长边线拉回。我们最初使用的四边形网格长宽比与最终我们得到的几乎等长的三角形网格是多么的接近。

   示意：一开始会了得到这个最好的效果，使用一个长宽比例为 1:1.73 的四边形。尝试使用一个相同比例的不同粗糙网格重新开始。

   最后的问题是三角形，因为我们还没有使用 Add Parallel Diagonals 工具进行分割。有很多方法可以完成，你可以删除它们然后使用两个三角面替换，手工处理是一件非常琐碎的事情但是是一个非常好的机会用来编写 RhinoScirpt 代码。如果对此很兴趣，可以阅读 Marko 编写的 EvoluteTools Sripiting Primer 一书，这里不再重述。

9. 你只需奥执行 Dual 细分就可以很轻松将三角形网格专为为六边形网格。[或你有更好的办法来制作六边形网格。]



9. 如何 充分利用 EvoluteTools

这一章主要向你提供一些使用 EvoluteTools 的诀窍与技巧。当然这里将要讲解的是最常用的方式与最优方法，而不是以一组特定范例与个案。

9.1 迭代过程

这本手册中的范例介绍是一个非常典型的线性过程，稍稍有一些误导。因为无法表现一个至关重要的部分，EvoluteTools 所使用的最有效的方法，即它是以迭代或循环的方式运行。

离散化一个自由曲面通常来说不是一个简单的任务，？而预先给去一个优化的结果非常困难（如果他很简单，还需要有优化这一步吗？）。EvoluteTools 的完美之处在于其非常的灵活且快速，灵活在于允许你在优化过程中调整优化参数或增加约束然后及时看到结果，允许你通过尝试与错误的实验，快速的反馈结果，当你第一次学习使用这个软件就能体验到特别真实。不要害怕重头开始且做一些新的尝试，第一个粗糙网格做的不好？再做一个然后重新编辑。

我有误导你吗？开始很好但中间遇到问题，不要太生气，还记得典型作业流程吗？实际操作中你并不会完全的套用这个典型作业流程，你常常会在细分与优化之间重复，我坦白：其实在细分之后我时常也会保留粗糙网格，因为粗糙网格与细分网格的父子参照关系对于调节造型非常用。

9.2 逻辑连接

下面的章节中，粗糙网格或父网格是用于细分时所选择的网格，新创建的网格会称之为细分网格或子网格。

细分后，两个网格之间存在一个逻辑连接。

什么意思？EvoluteTools 会在这两个网格之间维持一个连接，细分网格的顶点线性的依赖于粗糙网格的一个或多个顶点，如果父网格有做变形或其他编辑，子网格也会做相应的修改（根据原来创建细分模型时所使用的细分算法），这种关系很像曲线或曲面与其控制点的关系一样。

再次对已经细分的网格进行细分（使用相同或不同的算法），将会参数一个细分链，开启多重比例细分模型，仅支持使用 EvoluteTools 中提供的工具编辑细分网格连接关系。使用 Rhino 的工具删除细分网格中的顶点，将会从粗糙网格中自动复制这个网格。可以使用 etShowDependencies  指令来查看细分网格的连接关系。

重点: 优化仅作用于父网格的顶点，其他网格由细分算法生成与确定，他们是分开的。

首先，确定：当你有一个非常大且块面非常小的网格，优化程序可能会变慢因为要计算太多的顶点。如果你维持父网格与子网格之间的连接，优化程序运行会很快，因为仅需要考虑较少的顶点。

如果使用细分得到一个“不稳定”的物体时保留这种连接性也是很有用的。例如，由 Dual Edge 细分后得到的平直度，如果你移除了这种连接性，当执行优化程序后会移动顶点而丢失这种平直度。此外，如果保

留连接性，优化程序仅需移动父网格即可，改变后的子网格将仍然会是一个完美的 Dual Edge 细分网格且能保留应有平直度。

保留这这种连接性也会有一些负作用，如果父网格的顶点太少，不够优化程序运行所需要的输入，这样就无法得到一个逼近目标物体的优化结果。正是因为这样我才会在前面的范例中删除粗糙网格，或者你可以保留父级网格，然后使用 etDecoupleSubdividison 工具  删除连接性。

通常你需要处理第一个粗糙网格，并使用这种关系与后后面的网格关联，因为你的网格会越来越复杂。

忠实的读者，谢谢你能看到最后。我想你现在应该明白 EvoluteTools 是一个多么令人难以置信、功能丰富的小软件，虽然这本手册有近 60 多页，我也仅能介绍我所肤浅认知的层面，如何使用它所提供的特征来协调复杂几何的创建与实现。当然，这仅是一本入门手册，现在你应该考虑自己如何去深入学习。最后，对于如何成功的使用 EvoluteTools，我要给你 3 条法则：

1. 尝试
2. 尝试
3. 尝试

你一定会感到惊讶，你竟然会完整的读完，且也可以做复杂形体的优化。

对于这个教程你有任何的疑问，或是查找更多的 EvoluteTools 教程，请浏览 Evolute 网站：http://www.evolute.at/ 与 YouTube 频道 ：http://www.youtube.com/user/evolutegeometry

或联系我们的技术支持: support@evolute.at

谢谢阅读且希望你能喜欢 EvoluteTools！

Revision 1.0

2012 年 12 月 15 日

关于中文手册翻译

这是一本写的非常好的技术手册，由于时间仓促，且原文中不乏专业名称与用语，可能译文阅读起来会让你觉得很是晦涩。尽管原作者妙语连珠，将一些生僻的专业名称也口语化的恰到好处，但由于个人 E 文能力有限，可能无法让你体会到这样的阅读乐趣，在此表示歉意。

如有任何的翻译用词与技术性错误，竭诚欢迎来信指出，EvoluteTools 真是一个非常实用的工具，如有任何 EvoluteTools 软件的相关问题也欢迎与我联系，例如使用、培训以及其他的技术上的问题等。

Jessesn Chen
jessesn@shaper3d.cn
Shaper3d Studio

中文版，第一版

2012 年 12 月 26 日