幾何、藝術、計算機科學 ── 多面體展開的奇幻世界

張以潤／臺大電機工程系四年級學生
顏嗣鈞／臺大電機工程系教授

相信各位在小學時期的數學課或美勞課時都做過類似的事情：在西卡紙畫上某個多面體的展開圖後，將之剪下，再折成原本的多面體。估計一般人怎麼想都不會把「多面體展開」與「電機工程學」聯想在一起吧！但事實上，這個特別且有趣的事情在計算機科學領域當中可是一個研究議題，在理論上或應用上都有許多值得探討的地方。

多面體（polyhedral）是一個基本且重要的幾何概念，許多日常生活接觸到的物品都是多面體，如小時候常在玩的模型與積木。而某些藝術品，像是雕塑，也經常會包含許多複雜的多面體結構，因此，藝術家們也會關注幾何學上的議題。當我們要認識一種多面體時，一個常見的方式是去繪製它的展開圖（unfolding），所謂展開圖即是沿著多面體表面的某些邊剪開，展開後所形成平面上的圖形。早在 1525 年，一位德國藝術家 Albrecht Dürer 在他關於幾何學的著作 Underweysung der Messung [1] 中便大量地使用了展開圖的方式介紹了各種多面體，見圖 1。

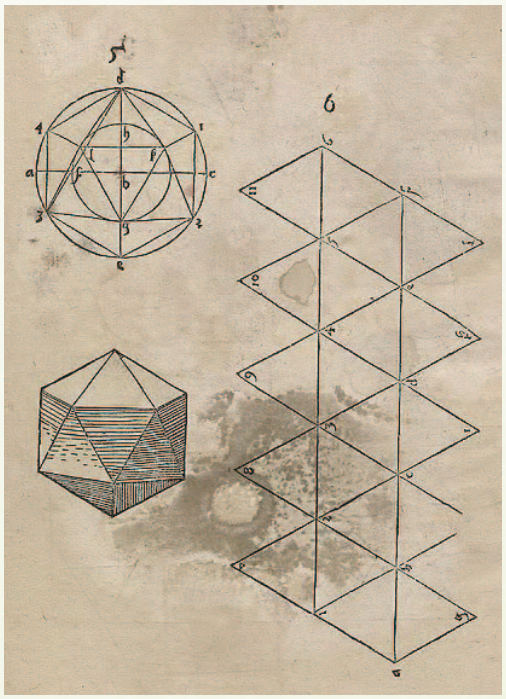


圖 1　Dürer 在他的著作 [1] 中使用展開圖介紹多面體。

回憶起繪製展開圖的過程中，我們往往為了方便性，會要求展開圖必須是連通的單一平面，而不是分散成許多區塊，這在攜帶上，或黏合的過程中都會比較方便。另外，為了要讓展開圖能畫在平面上，多面體展開後的圖形不可以有重疊的部分，見圖 2。因此，有人就開始思考一個問題：「是否所有多面體都能夠沿著某些邊剪開，展開成單一沒有重疊的平面圖形？」。顯然地這是不對的，因為圖 3 就給出了一個不可能被展開而沒有重疊的多面體的例子（可以試著建構看看其它例子，並不會很困難。）。從現在起，我們談的展開都是局限於僅能延著邊剪開，這樣的展開又稱做邊展開（edge unfolding）；而相對地，不對剪的位置做限制的展開則稱做一般展開（general unfolding）。若沒特別註明，以後講到的展開都是指邊展開。

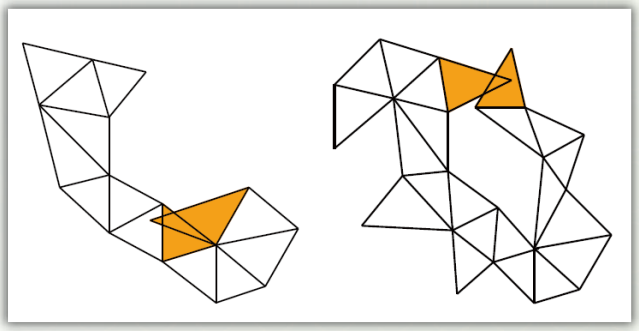


圖 2　橘色的部分即是重疊，易見無法畫在平面上，圖片出自 [4]。

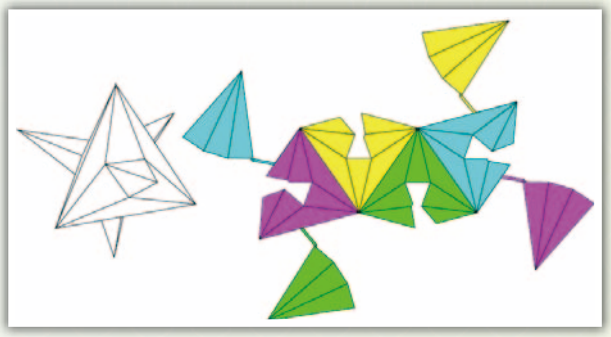


圖 3　左圖為一個不存在「邊展開」的多面體，如果一定要將之展開成平面的話，一定得把某些面剪開，如右圖為其「一般展開」，圖片出自 [2]。

直覺地想，一個多面體如果不要有太多凹凸的結構的話，就可能比較容易展開（各位可以思考看看怎麼樣的多面體會比較容易／不容易展開。）因此，在 1975 年時，一位數學家 Shepherd 就提出一個猜想如下：「所有的凸多面體都可以被展開。」雖然這個猜想看起來很單純，但難度卻相當高，至今只有很少數的特例被證明可以被展開。而另一方面，也沒有人能夠成功地建構出無法被展開的凸多面體。總而言之，這個猜想距離被解決尚有一大段距離，但大多數人相信它是對的。

在計算機科學中，一個很重要的概念是演算法（algorithm）。如果概略地地說的話，所謂演算法其實就是「能夠解決一個問題的具體方法」，我們會要求這個具體方法的每個步驟都是很明確的，不能有任何歧異，而且正確性是被保證的，例如大家在數學課中學過的輾轉相除法就是一套求最大公因數的演算法就是一個例子。因此，在多面體展開的議題中，從演算法的觀點來看的話，我們會想知道是否存在一套演算方法可以把任意的多面體展開，如果該多面體可以被展開的話。

回想一下以前小時候設計展開圖的做法，想必是沒什麼規則性地試一試，畫一畫，發現可行，就成功了（像這樣的做法不能稱之為演算法，因為步驟不明確，也沒辦法確保正確性。）因為我們遇到的多面體都很容易，所以大家都不會去想到要去設計一套演算法去做這件事情。但是，如果我們現在想要展開的多面體並不是像以往所面對的正立方體、正八面體般地簡單，而是個有著上千上百個面的複雜多面體時，就不太可能像這樣用亂試的方式輕易展開了。而且，在這樣複雜的情形中，用手算幾乎是不可能的，因此我們必須要用電腦程式去做，以加速運算。計算機科學之中的一個分支－計算幾何（computational geometry），即是專門研究像這種以演算法觀點去解幾何問題的領域，而多面體的展開自身就是其中一個研究議題。

除了先前提到理論上的問題外，把多面體展開究竟還能做些什麼呢？（各位可以想一想。）在計算機圖學 (computer graphics) 的領域中，我們會探討如何將各種我們有興趣的物體用電腦程式呈現，其中三維模型（3D model）是指三維中物體的表面的數學表示方法。現實上，我們有興趣的物體，例如動物，往往有著相當複雜的表面，因此它們的三維模型無可避免地會失真，所以我們只能去做近似。三維模型有很多種表示法，其中一種是多邊形模型（polygonal modeling/meshing），即是將物體的表面以多個多邊形近似，產生一個多面體的表面。可想而

知，如果要求失真越小，所需要的多邊形數量將會越多，即是產生的多面體會有很大量的面數。我們可以透過此法，用一個多面體去模擬我們感興趣的東西，像是兔子、手、飛機 ··· 等等。如果我們想要實際地在現實生活中做出我們感興趣東西的模型的話，利用上面所述的概念，我們可以造出它的多邊形模型，再來建構該模型的展開圖，這就是一個紙模型（paper craft）了，把它印下來，我們就可以實際地把它折回，做出來，圖 4 展示了一個例子。

圖 4　紙模型的範例：以 348 個面的多面體模擬一隻兔子，透過電腦程式展開後，在現實中把展開圖折回原本兔子，出自 [4]。

但很不幸地，在理論上，設計一套可以用來展開多面體的演算法並不簡單（各位可以試試看。），除了少數特例外，目前並沒有什麼好的演算法可以保證做到此事。就算是看似最簡單的凸多面體，我們了解的也很有限，Schlickenrieder 在他 1997 年的碩士論文 [3] 中設計了數十種演算法試圖展開凸多面體，結果都有失敗的例子，可見難度之高。許多科學家或工程師在解決問題時都會從日常生活中尋找點子，讓我們來看一個單純的想法：我們再削蘋果時，觀察到可以把蘋果的表皮拉成一個很長的帶子，這個過程很像是將蘋果的表面給展開，那我們是否可以用類似的手段去「削」一個凸多面體，使得產生的展開圖沒有重疊呢？又是很不幸地，答案是不行的，從實驗結果中可發現在開頭和結尾處往往很難避免重疊，見圖 5。雖然失敗了，不過類似的概念確實是可以應用於展開某些特殊的多面體，像是一種名為 orthogonal terrains 的多面體類型就可以被一層一層地剝皮展開，見圖 6 所示。

圖 5　嘗試用削蘋果的概念展開一個凸多面體，紅圈處有重疊。

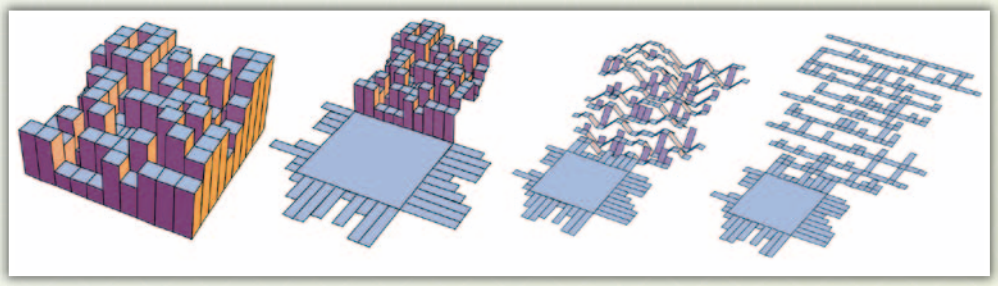


圖 6　展開「orthogonal terrains」，圖片出自 [5]。

以下我們會介紹展開多面體的一些手段，在此之前，會需要一些預備知識。在計算機科學中，我們常會用到圖（graph）做為許多事物的模型，所謂圖 G = (V, E) 就是一些點（vertex）的集合 V 與邊（edge）的集合 E，定義邊為兩點的連結，例如 e = (u, v) 表示了 e 為兩點 u 與 v 的連邊。一般來說，這些點與邊是不具有幾何上的意義的，根據不同用途，我們可以賦予它們不同的意義。在電子設計自動化（Electronic Design Automation）的領域中，一個圖中的點可能代表了邏輯閘，而邊代表了線路；對於社交網路（Social network）的研究者來說，點可能表示了 Facebook 上的使用者，而兩點有無連邊則表示它們是否將對方加為好友。

對於某多面體 P，如果我們選取 V 為他的點集，而 E 為他的邊集的話，就可以形成一個圖 G = (V, E)。類似地，若把點集取成 P 上的面，而兩面相連則表示它們連邊，則形成的圖 G* 稱做 G 的對偶圖（dual graph）。定義路徑（path）為一連串的點 $v_1v_2\cdots v_k$ 使得任兩點 v_i, v_{i+1} 滿足 $1 \le i < k$ 都有邊連結之。而類似地，如果進一步要求 v_k, v_1 兩點之間也有邊連結的話，則為圈（cycle）。我們說一個圖中兩點 u, v 是連通（connected）的意思就是說存在一條路徑使得 u, v 為它的兩端點。如果一個圖中任意的兩點都連通的話，我們說這個圖是一個連通圖（connected graph）。而如果兩個圖 $G_1 = (V_1, E_1)$ 與 $G_2 = (V_2, E_2)$ 滿足 $E_1 \subseteq E_2$ 且 $V_1 \subseteq V_2$ 的話，我們說 G_1 為 G_2 的子圖（subgraph）。我們定義樹（tree）為沒有圈的連通圖，而一個連通圖 G 的生成樹（spanning tree）為該圖的一個子圖 G′ 使得它為一個樹，且 G′ 的點集與 G 相同。

如果該多面體可以被展開的話，那些被我們剪開的邊所形成的子圖 G′ 必為原圖 G 的一個生成樹，我們稱之為剪樹（cut-tree）。另一方面，我們考慮對偶圖 G*，如果 G* 的子圖 G″ 為那些展開後未被剪開的邊，則 G″ 也會是 G* 的一個生成樹，稱做鄰樹（adjacency-tree）。或許上列大量的定義會讓讀者感到些許混亂，可以見下圖的例子以了解它們在實際上的感覺。

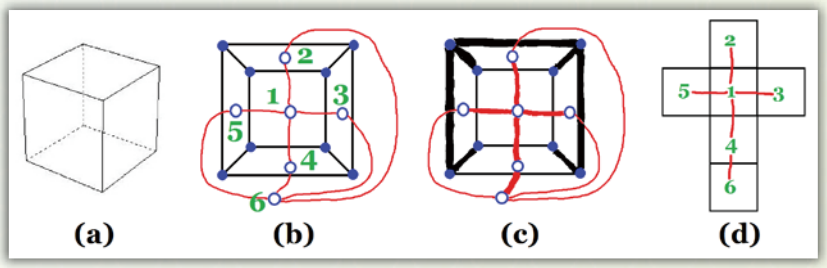


圖 7　(a) 為一個正立方體，現今想像把它立在紙上，如 (b), (c)，為方便起見我們把外圍當做被壓住的那一面。(b) 中的黑線和實心圓圈表示了由該立方體的點和邊所構成的圖 G，而紅線和空心圓圈則表示了它的對偶圖 G*。(c) 中的黑粗線為 G 的一個剪樹，而紅粗線則是 G* 中相對應的鄰樹。當我們沿著剪樹上的邊剪開後，鄰樹恰表示了展開圖中各個面的相鄰情形，如 (d) 中所示。

如此一來，我們只要試該多面體所對應的圖 G 或對偶圖 G* 的各種生成樹，看它們展開後所形成的展開圖是否有重疊，即可知道這是不是一個沒有重疊的展開。但是，一個圖可以有非常多種的生成樹，故我們沒能指望暴力地去試完所有可能。而從各種實驗數據中亦顯示，隨著面數的提升，隨機展開成功的機率會快速降低。我們無可避免地需要使用一些啟發式（heuristic），即是一些幫助我們下決策的規則，去猜怎麼樣的生成樹會比較容易成為一個沒有重疊的展開。我們無法保證依賴啟發式必能產生沒有重疊的展開，因此以下介紹的方法都沒有正確性的保證。

現在我們就可以來看 Schlickenrieder 的碩士論文中的一個演算法，名為 steepest-edge-algorithm。在他的實驗報告中，此演算法產開的展開圖只有 1.2% 的展開有重疊，其餘的 98.8% 都成功了。這個啟發式的概念為選取針對某個方向 c「最直」的那些邊，並且根據所形成的生成樹剪開來展開多面體。具體來說，就是對除了 v+（在 c 方向上最高的點）以外的所有的點 v，在所有由 v 連出去且比 v 還要高的邊之中，選一個與 c 夾角最小的，把他加到 T 中。可以證明如此選出的 T 為一個生成樹，過程在此省略。可以預期，如此地展開的結果會很像是把整個多面體從 v+ 撥開，如同開花般的感覺，見圖 8。而實際上該演算法產生的展開圖就是如此，見圖 9 漂亮的例子。

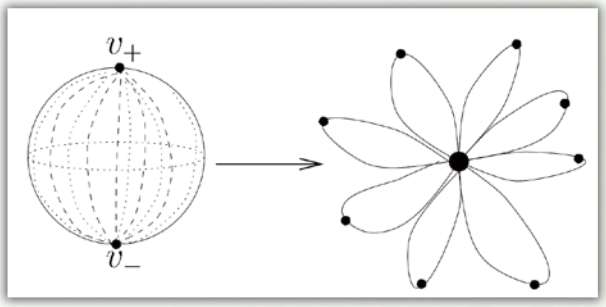


圖 8　steepest-edge-algorithm 的概念，出自 [3]。

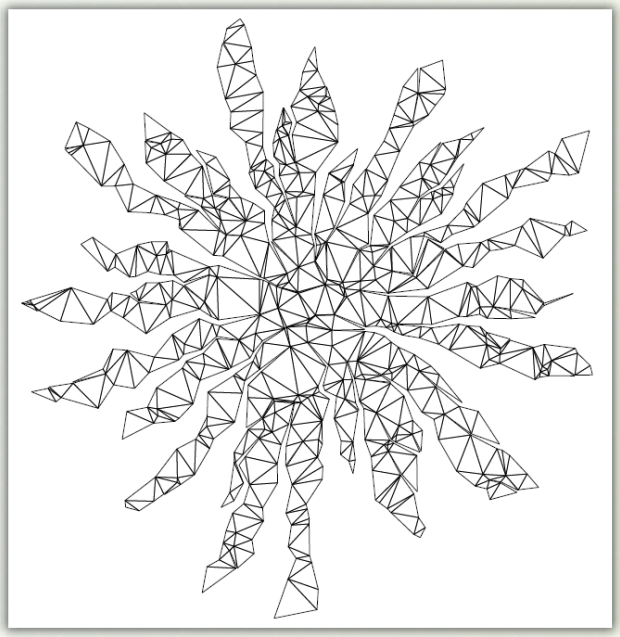


圖 9　一個運用 steepest-edge-algorithm 展開的例子，出自 [3]。

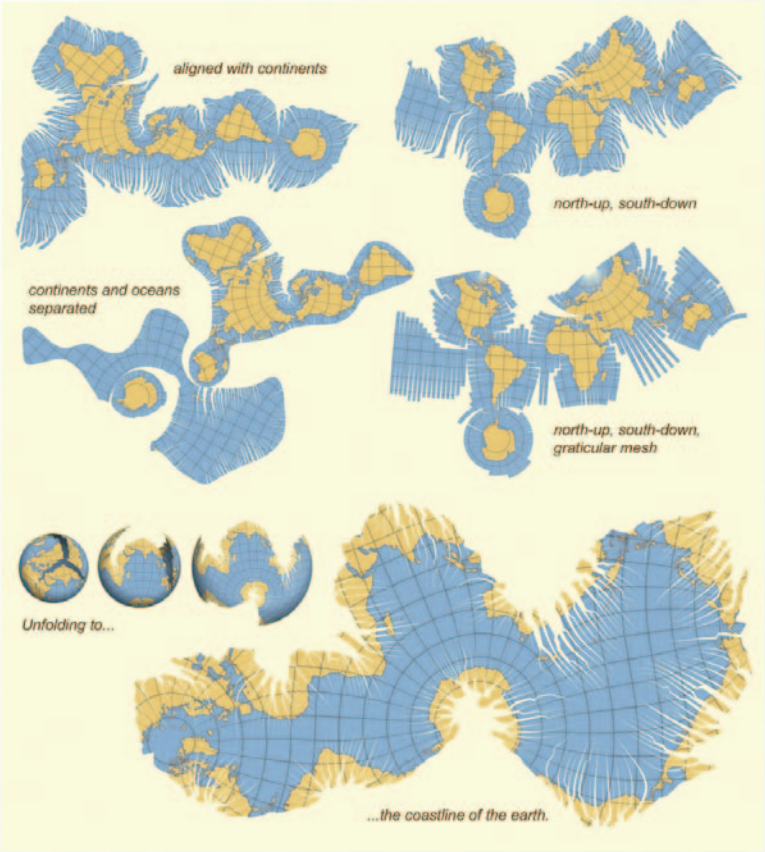
至今，這套啟發式演算法依然是目前所知對於凸多面體展開最佳的方法，但是它在應用上的用途依然很局限，因為我們感興趣的多面體往往都不是凸多面體（steepest-edge-algorithm 對於非凸面體效果一般來說並不佳，且遠不及其在凸多面體上的表現，為什麼？），這時，要找到一套很一般的啟發式就不容易。回想我們說過所有的展開都是一個生成樹，而若給定一個生成樹，其對應的展開圖有可能會有重疊，故我們可以依照生成樹重疊的嚴重性（像是重疊的面數）給該生成樹一個分數。我們原本的問題就可以轉化成一個最佳化問題（optimization problem），即是在所有的生成樹中找到一個生成樹使得它的分數最低，此時如果這個多面體是有辦法被展開的，那麼分數最低的生成樹的分數就會是 0，即是沒有任何重疊。

但先前有提過，由於一個圖可以有著相當大量的生成樹，故全面性的搜索是不可行的。此時，我們可以採取局部搜尋（local search），即是不斷地修改目前搜尋到的結果。我們簡介一種最單純的局部搜尋方法，稱做爬山演算法（hill climbing algorithm）。我們可以將想要搜尋的空間想成一個地面上的點，在此就是指所有的生成樹的集合所形成的空間。對於不同生成樹，根據其展開後的重疊程度，讓越輕微者所對應的點高度越高。定義兩個生成樹相鄰若且唯若它們可以從對方刪掉一邊後再加入另外一邊而建構出來，即是兩生成樹的差異恰只有兩邊。於是該平面就會長得像是一些丘陵，而最高的那些地方就對應著展開後沒有重疊的生成樹們，即是我們搜尋的目標。

於是我們只要先隨機選取一個生成樹 T，然後試著修改它，即是刪掉一邊後再加入另外一邊，使得新的圖 T′ 亦為生成樹，我們看看 T′ 展開後重疊的程度是否比 T 輕微，是的話就以 T′ 取代 T，否的話就試試其它修改方式。如此一來，最後就有可能會達到最高點，得到一個沒有重疊的展開方式。但這只是「有可能」而已，各位應該可以輕易地發現，因為這套方法並沒有下山的機制（是否應該在某些情形下允許下山，即是允許 T′ 的重疊程度比 T 嚴重？許多局部搜尋演算法是有下山機制的。），如果我們不幸地走到了一個不夠高的山的話，它的山頂對應的生成樹展開可能還是會有重疊的，此時就得重新搜尋一次。不過實驗結果告訴我們，只要輸入的多面體不要太複雜的話，簡單的爬山演算法其實效果還滿不錯的，雖然它沒有辦法保證給出正確答案。

注意到我們在生成樹的選取上是純粹隨機的，因此這套方法應該有頗大的改進空間，各位可以想想該怎麼改進。舉例來說，或許可以在生成樹的選取上直接避免一些顯然有問題的結構，例如圖 2 中的左圖右下方六個共點的面與該點夾角的和已經超過 360 度了，所以我們知道一定至少要再剪一邊掉。

相信各位在地理課時學過許多地圖的投影方式吧！如圓柱投影、圓錐投影等等。最後，讓我們來看一個創新的應用，這是一個將多面體展開運用在地圖投影上的方法，把「多面體展開」與「地圖投影」兩個看似不相關的概念巧妙地連結在一起。我們可以將地球以一個面數相當大的多面體（myriahedral）去近似，然後再把它展開，得到的展開圖就是一個平面的地圖。可見這樣的做法在面數很多且每面都夠小時，地圖的失真可以控制到很小。我們可以根據不同的多面體表面的建構方法，以及生成樹的選取方法，構造出各種有趣且富有意義的地圖，如圖 10。



多面體展開尚有許多應用，例如一般展開中的星形展開（star unfolding）與多面體表面的最短路徑（shortest path）計算有著密切關聯；若不限展開成單一平面的話，也有文獻探討透過展開多面體去做三維模型資料壓縮（data compression）的方法。在計算機科學之中，多面體展開，或者計算幾何、計算機圖學等各種子領域裡，在理論或應用上都有著許多有趣的問題，等著各位去發現和解決！

圖 10　透過不同展開的方法，產生有趣的世界地圖，出自 [6]。

References

[1] Albrecht Dürer. Unterweysung der Messung mit dem Zirkel un Richtscheyt in Linien Ebnen und Gantzen Corporen, 1525.

[2] E. D. Demaine and J. O'Rourke. A survey of folding and unfolding in computational geometry. In J. E. Goodman, J. Pach, and E. Welzl, editors, Mathematics Sciences Research Institute Publications, volume 52, pages 167-211. Cambridge University Press, 2005.

[3] W. Schlickenrieder. Nets of Polyhedra. Master's Thesis, Technische Universität Berlin, June 1997.

[4] Shigeo Takahashi, Hsiang-Yun Wu, Seow Hui Saw, Chun-Cheng Lin, and Hsu-Chun Yen, "Optimized Topological Surgery for Unfolding 3D Meshes," in Computer Graphics Forum, (special issue for Pacific Graphics 2011), Vol. 30, No. 7, pp. 2077-2086, 2011.

[5] Joseph O'Rourke. Unfolding orthogonal terrains. Technical Report 084, Smith College, July 2007.

[6] J.J. van Wijk. Unfolding the Earth: Myriahedral Projections. The Cartographic Journal, 45(1), pp. 32-42, 2008.