Algoritma Untuk Penentuan *Seams* dalam Proses *UV-Unwrapping*

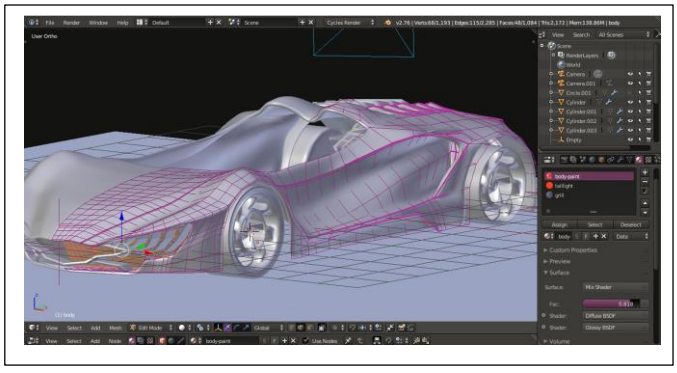
Muhammad Reifiza 13514103
Program Studi Teknik Informatika
STEI ITB
Bandung, Indonesia
13514103@std.stei.itb.ac.id

***Abstrak*—Dunia modern tidak terlepas dari kebutuhan pemodelan dalam bidang tiga dimensi untuk memenuhi berbagai kebutuhan, misalnya dalam bidang hiburan dan *digital printing*. Salah satu proses yang terdapat dalam pemodelan adalah *UV-Unwrapping* yang bisa dilakukan manual maupun automatis. Biasanya, *UV-Unwrapping* membutuhkan definisi *seams* (lipatan) supaya hasilnya cukup baik. Dalam makalah ini penulis akan menjelaskan beberapa algoritma untuk penentuan *seams* dalam proses *UV-Unwrapping*.**

Keywords—algoritma, graf, UV-Unwrapping, 3D

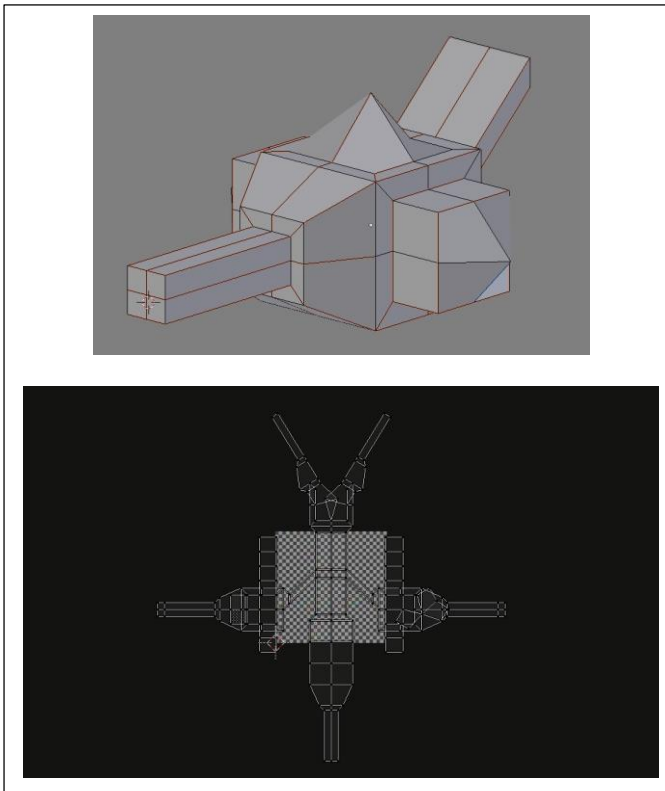
I. PENDAHULUAN

Dahulu sebelum era 90-an, penelitian dalam bidang grafika komputer atau seni digital masih terbilang jarang, mengingat pada saat itu komputer yang ada belum secepat sekarang (belum *feasible*). Media seni tradisional seperti kertas dan pensil untuk menggambar atau pahat untuk mematung adalah media utama saat itu. Seiring perkembangan jaman, komputer berevolusi menjadi semakin cepat sehingga penelitian dan implementasinya dalam bidang seni digital menjadi semakin layak dan semakin sering dilakukan peningkatan dari implementasi sebelumnya. Oleh karena itu, saat ini istilah *digital artist* atau seniman digital semakin dikenal masyarakat global, karena pengaruh mereka dalam evolusi pemikiran desain (*design thinking)* masyarakat begitu besar. Hal ini menyebabkan banyak orang yang tertarik dengan bidang ini, baik di dalam pengembangannya maupun menjadi penggunanya.



Gambar 1. Contoh Model yang dibuat dalam aplikasi Blender (Dokumen pribadi penulis)

Salah satu cabang dari seni digital adalah seni rupa digital tiga dimensi (selanjutnya disebut grafis 3D). Untuk semua cabang dalam grafis 3D (animasi, pembuatan maket, *game development*, dan lain-lain), ada sebuah proses yang selalu ada, yaitu pembuatan model 3D (*modelling*).



Gambar 2. Contoh model dengan *seams* (warna merah kecoklatan) (atas) beserta hasil *UV-Unwrap*-nya (bawah).

Proses *UV-Unwrapping* (atau terkadang disebut *UV-Mapping)* adalah proses yang cukup vital dalam rangkaian

proses *modelling*. Setelah sebuah model jadi, pemodel harus melakukan *UV-Unwrapping* model tersebut untuk menempelkan sebuah gambar tekstur untuk memberikan kesan *liveness* terhadap model tersebut, namun pemodel tidak harus memberikan sebuah gambar tekstur setelah melakukan *UV-Unwrapping* pada model tersebut. Setiap paket aplikasi grafis 3D menawarkan cara masing-masing untuk melakukan *UV-Unwrapping*, namun umumnya proses ini bisa dilakukan secara manual maupun automatis. Selain itu, model sekaligus peta UV (*UV Map*, hasil dari proses *UV-Unwrapping)* bisa di-ekspor-impor antar paket aplikasi grafis 3D. Oleh karena itu, jika sebuah proyek seni digital melibatkan lebih dari satu orang, *UV Map* model yang sudah didefinisikan oleh pemodel merupakan *workflow* yang lumrah.

Prinsip dari proses *UV-Unwrapping* adalah menentukan jaring-jaring dari model tersebut. Sebuah jaring-jaring bangun adalah sebuah graf planar yang membangun bangun tersebut, seperti yang sudah diajarkan di sekolah dasar. Untuk bangun sederhana semisal prisma dan limas, menentukan jaring-jaringnya adalah hal yang mudah. Namun, bangun-bangun sesederhana itu lebih jarang digunakan daripada bangun yang memiliki topologi lebih kompleks.

Dalam penulisan makalah ini, penulis menggunakan Blender sebagai aplikasi pendukung. Dalam aplikasi ini, pemodel dapat mendefinisikan *seams* (lipatan) yang fungsinya untuk mendefinisikan bagian mana yang akan di"gunting" supaya terbentuk jaring-jaring bangun tersebut. Beberapa aplikasi grafis 3D lain seperti Autodesk 3DS Max juga mempunyai fitur serupa.

Masalahnya, terkadang pemodel, terlebih pemula, bingung sisi (lihat definisi di bagian II) mana saja yang harus dijadikan *seam* supaya *UV-map* yang dihasilkan baik. Sebuah *UV-map* dikatakan baik apabila *UV-map* tersebut memenuhi kriteria berikut ini.

- Merupakan sebuah graf planar. Ini merupakan konsep dari jaring-jaring bangun.
- Ketika digunakan untuk menempelkan gambar ke model, gambar tidak terdistorsi dan memiliki orientasi kemiringan yang sama dalam sebuah muka (lihat definisi di bagian II) dan memiliki proporsi yang seimbang untuk keseluruhan muka pada model.

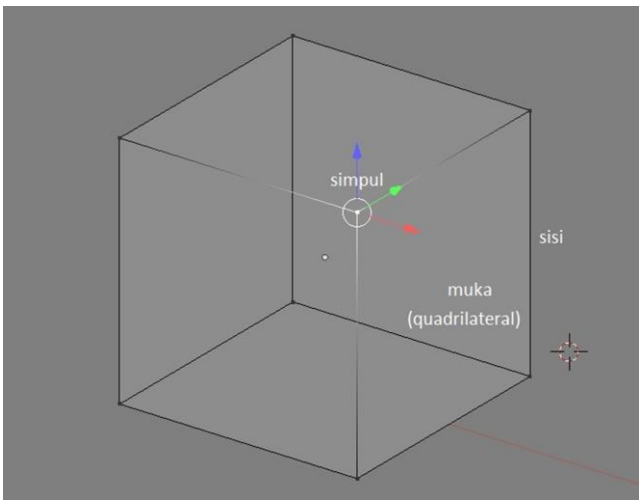
Semakin kompleks sebuah model, semakin rumit pendefinisian *seams* sehingga terbentuk *UV-map* yang baik. Seringkali pemodel dipusingkan dengan penentuan *seams* karena penentuan *seams* yang salah bisa mengakibatkan gambar yang ditempelkan pada bangun menjadi tidak enak dipandang. Berangkat dari masalah ini penulis bermaksud untuk memberikan ide algoritma yang dapat diterapkan untuk penentuan *seams*, baik untuk diterapkan langsung dengan cara manual oleh pemodel maupun menjadi sketsa algoritma alternatif automisasi proses *UV-unwrapping*, sehingga tercipta sebuah *UV-map* yang baik. Selain itu, penulis berkonsentrasi terhadap proses terbentuknya *UV-map* yang menimbulkan *island* (muka terpisah dari muka lainnya) seminimal mungkin karena alasan kebanyakan pemodel masih menggunakan cara manual adalah menghindari terbentuknya *island* yang terlalu banyak.

II. Definisi dan Teori

Berikut ini adalah jabaran definisi dan teori yang penulis gunakan di makalah ini. Penulis mengasumsikan pembaca telah mengetahui / memahami konsep graf.

A. Definisi Model

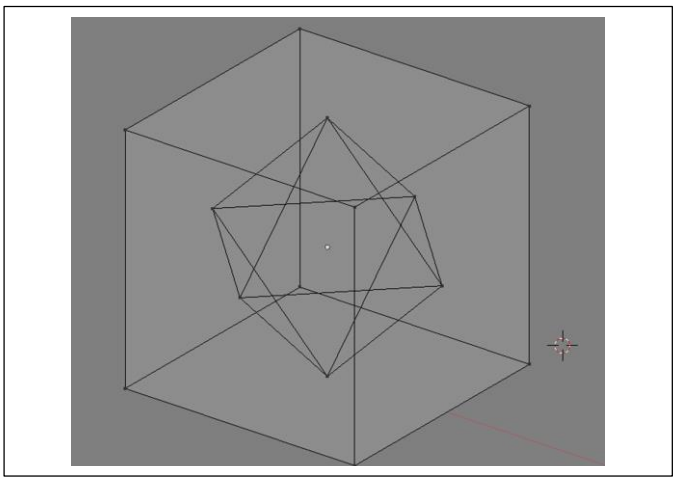
Sebuah model adalah sebuah set *mesh* yang terdiri dari sebuah *mesh* atau lebih. Seperti halnya sebuah graf, sebuah *mesh* adalah sebuah set yang terdiri atas *vertex* (simpul), *edge* (sisi), dan *face* (muka). Dua buah simpul dapat membentuk sebuah sisi dan tiga atau lebih sisi dapat membentuk sebuah muka. Sebuah muka bisa jadi triangular bila dibentuk oleh tiga sisi, quadrilateral bila dibentuk oleh empat sisi, atau poligonal bila dibentuk lebih dari empat sisi. Triangular dan quadrilateral lebih umum digunakan dibandingkan poligonal dengan sisi lebih dari empat karena bentuk poligonal mungkin menghasilkan topologi yang aneh. Dalam makalah ini, penulis akan membatasi model menjadi hanya mengandung sebuah *mesh*, juga penulis akan menggunakan istilah model, *mesh*, bangun secara bergantian untuk merujuk hal yang sama.



Gambar 3. Ilustrasi sebuah model.

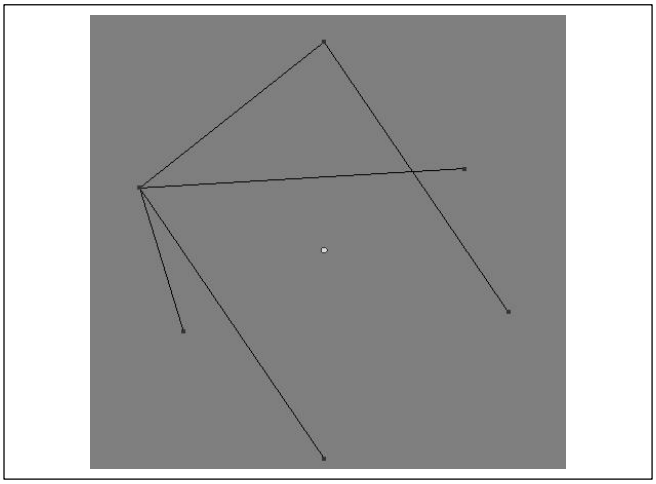
B. Definisi Graf Keterhubungan dan Skeletal Bangun

Didefinisikan graf keterhubungan, yaitu sebuah graf yang dibuat dengan menghubungkan titik pusat setiap muka bangun



Gambar 4. Ilustrasi graf keterhubungan (berada di dalam kubus).

yang bertetangga. Maksud dari muka yang bertetangga adalah muka-muka yang memiliki minimal sebuah sisi persekutuan. Graf ini mungkin akan mengandung banyak upagraf yang siklik. Sebuah skeletal bangun adalah sebuah pohon yang merupakan upagraf dari graf keterhubungan yang memuat seluruh simpul *mesh* model. Karena skeletal adalah sebuah upagraf dari graf yang lebih besar, maka skeletal suatu bangun bisa jadi lebih dari satu. Konsep graf keterhubungan dan skeletal ini yang menjadi fondasi salah satu algoritma yang akan penulis bahas.



Gambar 5. Salah satu graf skeletal dari graf keterhubungan pada gambar 4

C. Definisi Pohon Status

Dalam beberapa algorima yang berbasiskan graf, pohon status sering dilibatkan. Maksud dari algoritma berbasiskan graf adalah algoritma yang menggunakan graf sebagai metode pencapaian solusi dalam pemecahan masalah. Pohon status adalah sebuah graf pohon yang simpulnya adalah status / kondisi masalah saat pendekatan pada suatu iterasi. Pohon ini bisa diimplementasikan dengan struktur data yang konkrit (*array*, *list*, dll.) atau dengan struktur yang abstrak (fungsi rekursif).

D. Beberapa Algoritma Berbasiskan Graf

Karena algoritma yang berbasiskan graf ada banyak, penulis hanya menjabarkan beberapa diantaranya yang akan digunakan sebagai ide dasar algoritma untuk penentuan *seams*.

1) Algoritma Breadth First Search (BFS)

Algoritma BFS adalah algoritma yang mengiterasi simpul pohon status secara melebar. Maksudnya, iterasi dilakukan dimulai dari simpul status *parent* (bapak) dengan memasukkan semua kemungkinan status yang dapat dicapai dari simpul tersebut ke dalam *queue*. Iterasi dilakukan dari status pertama yang berada dalam *queue* tersebut, sehingga terciptalah anak pertama dari simpul status akar. Proses ini dilakukan terus menerus hingga dicapai solusi.

2) Algoritma Depth First Search (DFS)

Algoritma DFS adalah algoritma yang mengiterasi simpul pohon status secara mendalam. Maksudnya iterasi dilakukan dimulai dari simpul status *parent* dan langsung dilanjutkan dengan kemungkinan status yang dapat dicapai. Proses ini dilanjutkan hingga : (a) tidak ada lagi kemungkinan status *feasible* yang dapat dicapai dari status saat ini; (b) solusi tercapai. Jika kondisi (a) yang tercapai, maka harus dilakukan perunutan balik ke status sebelumnya yang mungkin masih ada status *feasible* yang dapat dicapai.

E. Lainnya

Dalam penulisan makalah ini, penulis menggunakan aplikasi Blender yang merupakan *free and open source software*. Beberapa aplikasi mungkin tidak memiliki fitur untuk membebaskan penggunanya menentukan *seams* dalam proses *UV-Unwrapping*, seperti Autodesk Maya. Aplikasi lainnya mungkin menyebut *seams* dengan nama yang berbeda, namun mengacu pada hal yang sama.

III. ALGORITMA

Ada beberapa algoritma yang dapat digunakan untuk penentuan *seams* sedemikian sehingga tercipta sebuah *UV Map* yang baik dan menimbulkan *island* paling minimal. Berikut ini penulis jabarkan algoritma-algoritma tersebut.

A. Algoritma Naif

Algoritma ini yang paling mudah terpikirkan. Langkah-langkahny adalah sebagai berikut.

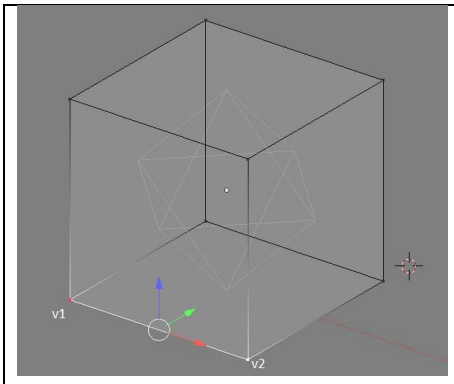
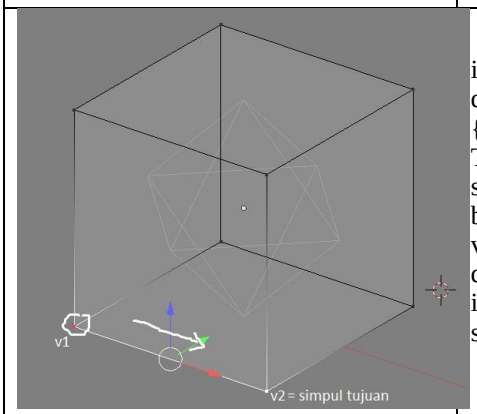
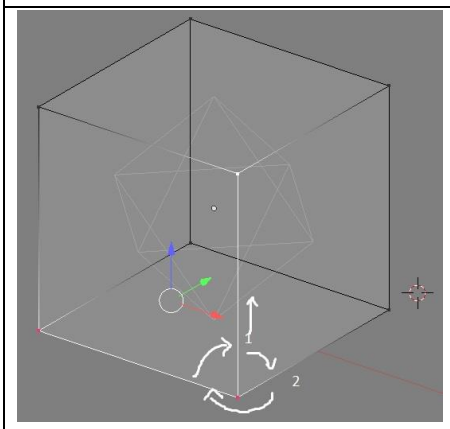
- Buat seluruh kombinasi sisi yang mungkin dalam sebuah bangun menjadi *seams*.
- Periksa *UV Map* yang dihasilkan dari proses *UV-Unwrapping* dengan memanfaatkan *seams* yang dihasilkan dari tahap sebelumnya satu per satu dengan tekun dan sabar sehingga ditemukan kombinasi yang menghasilkan *UV Map* paling memuaskan.

- Selesai.

Namun, algoritma ini sangat tidak mungkin diimplementasikan. Algoritma ini berjalan dalam $O(n!)$, dengan n menyatakan banyaknya sisi pada model. Tentunya algoritma ini adalah algoritma yang sangat lama. Untuk penerapan manual, yang mungkin terjadi adalah pemodel hilang kesabaran dan kehilangan waktu. Untuk penerapan terkomputerisasi tetap saja lama. Algoritma ini hanya layak jika sisi bangun jumlahnya sedikit. Kenyataannya, pemodel sering dihadapkan dengan bangun yang memiliki sisi sampai ratus-ribuan bahkan sampai puluh-jutaan. Oleh karena itu, penulis menyertakan beberapa alternatif algoritma.

B. Alternatif I

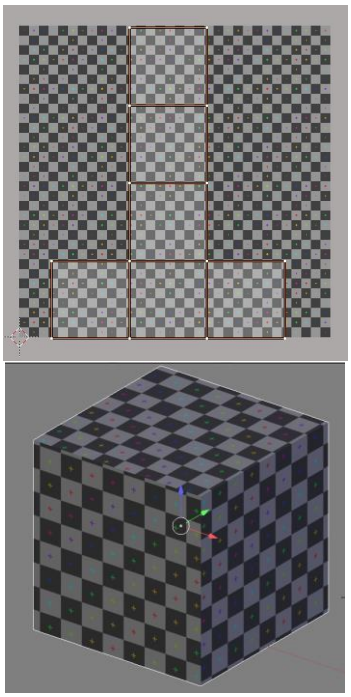
Algoritma ini adalah implementasi langsung dari algoritma DFS. Berikut ini adalah ilustrasi sekaligus penjabaran dari algoritma ini.

	Pilih salah satu sisi sembarang yang berada di *mesh* model.
	Misalkan sisi itu bersisian dengan simpul {v_1, v_2}. Tentukan arah simpul berikutnya (bisa v_1 maupun v_2) dan sebut simpul ini sebagai simpul tujuan.
	Jika simpul tujuan memiliki cabang ke simpul lain, maka pilih cabang pertama yang berada di dalam aturan berikut : *clockwise* atau *counter—lockwise* dan belum pernah dipilih sebelumnya. Jika ternyata pemilihan cabang menyebabkan graf siklis, pilih cabang berikutnya
Ulangi terus proses ini dari sisi cabang terpilih sesuai kaidah DFS. Berhenti jika sudah tidak ada kemungkinan lain. (solusi)	

Seperti algoritma naif, algoritma ini hanya cocok untuk *mesh* model yang memiliki muka sedikit, namun penerapan algoritma ini mungkin dilakukan secara manual oleh pemodel karena jauh lebih cepat dari algoritma naif. Alasannya adalah algoritma ini *immature*, masih banyak kemungkinan lain yang belum dipertimbangkan dalam algoritma ini, seperti berikut ini.

- Sebuah bangun mungkin memiliki dua atau lebih muka yang memiliki orientasi normal sama atau mendekati sama (menghadap ke arah yang sama) atau saling berlawanan / mendekati saling berlawanan, misalnya terjadi karena *mesh* dilakukan subdivisi berulang kali. Jika algoritma ini diterapkan, hasil dari *UV-Unwrapping* akan menjadi aneh.
- Sebuah bangun mungkin diterapkan *face inset* berulang-ulang, yaitu membuat muka baru di muka yang telah ada. Seharusnya, pemilihan *seams* yang baik mempertimbangkan kehadiran adanya *inset* ini, namun dengan algoritma ini hal ini mungkin terabaikan sehingga graf yang dihasilkan dari proses *UV-Unwrapping* mungkin menjadi graf bukan planar.
- Bergantung dari tahapan 1 dan 2, pemilihan sisi yang menjadi *seams* mungkin menjadi sangat tidak proporsional dengan banyaknya sisi yang ada dalam model. (misalnya sisi terpilih hanya ada 10 sedangkan jumlah sisi seluruhnya ada 50). Hal ini memungkinkan graf hasil *UV-Unwrapping* menjadi graf bukan planar.

Dengan berbagai masalah di atas, algoritma ini hanya cocok digunakan untuk *quick prototyping* oleh pemodel tanpa



Gambar 6. Hasil dari algoritma alternatif I

program, karena hasil akhirnya akan melibatkan pemodel untuk dilakukan perbaikan.

C. Alternatif II

Sebelum penulis menjelaskan tentang algoritma ini, ada baiknya penulis menjelaskan terlebih dahulu dasar pemikiran algoritma ini.

1) Dasar Pemikiran

Setiap jaring-jaring bangun memiliki banyak muka yang sama dengan bangun yang dibentuknya. Setiap muka yang ada di jaring-jaring bangun yang bertetangga dapat dihubungkan dengan sebuah garis. Himpunan garis-garis ini membentuk sebuah graf. Graf ini merupakan salah satu dari graf skeletal dari bangun yang dibentuknya. Sebaliknya, tiap graf skeletal yang merupakan subset dari graf keterhubungan bangun merupakan graf keterhubungan dari jaring-jaring bangun yang membentuknya. Oleh karena itu, jaring-jaring bangun yang membentuknya bisa lebih dari satu.

Setiap graf keterhubungan jaring-jaring bangun pasti memiliki sifat-sifat berikut.

1. Merupakan graf planar dan sederhana.
2. Tidak memiliki upagraf yang siklis.
3. Memiliki semua simpul yang ada di graf keterhubungan bangun yang dibentuknya dengan jumlah sisi graf keterhubungan jaring-jaring $\leq$ jumlah sisi graf keterhubungan graf.

Berdasarkan dasar pemikiran ini dapat dibentuk sebuah algoritma yang dapat memilih sisi bangun pada model menjadi *seams* sedemikian sehingga hasil dari *UV Map* yang dihasilkan cukup baik dan tidak memiliki *island* pada kebanyakan kasus.

2) Konstruksi Algoritma

Sebuah skeletal bangun dapat dibentuk dari graf keterhubungan bangun dengan memilih beberapa sisi dari graf keterhubungan bangun sedemikian sehingga terbentuk pohon merentang yang merupakan upagraf dari graf tersebut. Oleh karena itu, algoritma ini butuh *preprocessing* membentuk graf keterhubungan dari bangun yang akan dilakuka *UV-Unwrap*. Berikut ini adalah pseudocode untuk membangun graf keterhubungan tersebut.

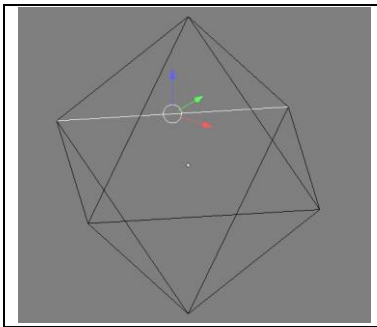
```
PROCEDURE CreateConnectingGraph(INPUT m : mesh;
            OUTPUT g : graph)
VARIABLE
     s: queue of face
     F: face
ALGORITHM BEGIN
FOR every faces in m DO
     insert central vertex to g
choose a face in m as F
push F to queue
WHILE (queue not empty) DO
     pop F from queue
     FOR every edges in F DO
            IF (there's another face with this
            edge and there's no connecting edges
            between     corresponding     central
            vertices in g) THEN
                  push face to queue
                  create     edges     between
                     corresponding vertices in g
            ENDIF
ENDWHILE
ALGORITHM END.
```

Jika diimplementasikan dengan baik dan benar, algoritma di atas seharusnya berjalan dalam $O(n)$, dengan n menyatakan banyaknya muka dalam sebuah *mesh* model.

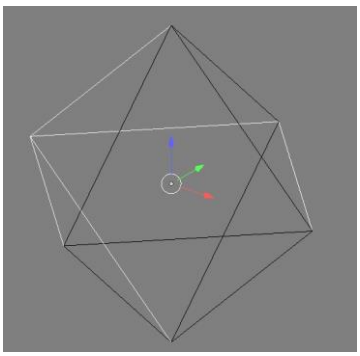
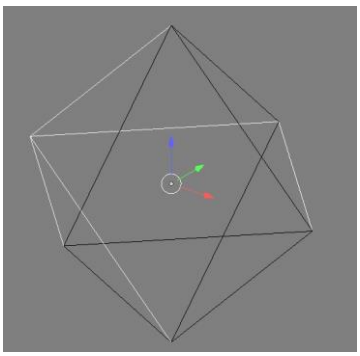
Setelah dibuat graf keterhubungan dari model yang akan dilakukan *UV-Unwrap*, barulah dibentuk satu graf skeletal dari graf tersebut. Ada banyak cara untuk membangun graf skeletal, namun di sini penulis akan menjabarkan algoritma yang penulis gunakan. Nantinya, setelah graf skeletal terbentuk, barulah dilakukan interpretasi sisi mana saja yang dijadikan *seam* untuk proses *UV-Unwrapping*.

Algoritma yang digunakan sebenarnya adalah modifikasi dari algoritma BFS. Gambar graf yang disediakan adalah contoh graf sederhana karena hanya untuk pengilustrasian

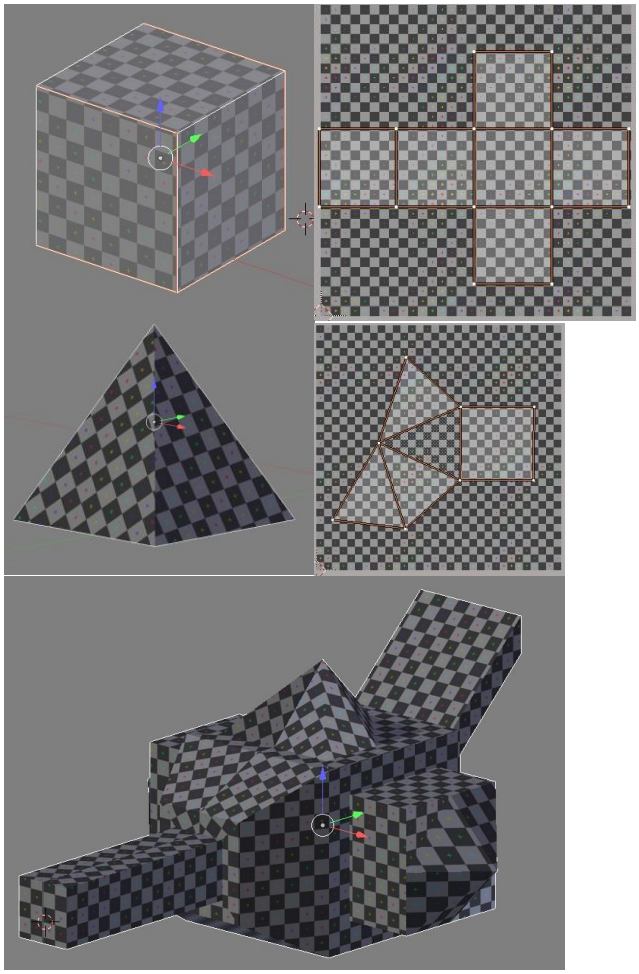
.



Pilih salah satu sisi sembarang dalam graf keterhubungan G. Kita sebut sisi terpilih adalah S dengan simpul yang bersisian masing-masing adalah v_1 dan v_2.

	Pilih seluruh sisi yang bersisian dengan v_1 dan sisi yang bersisian dengan v_2 kecuali sisi S dan sisi yang menyebabkan graf yang telah terbangun mengandung upagraf siklis. Seluruh simpul dari sisi-sisi yang telah diidentifikasi dimasukkan ke dalam sebuah tumpukan (*stack*) simpul SV.
	*Pop* sebuah simpul dari SV dan sebut saja simpul ini V_1. Pilih seluruh sisi yang bersisian dengan V_1 kecuali sisi yang pernah dipilih sebelumnya atau sisi yang menyebabkan graf yang telah dibuat mengandung upagraf siklis. Seluruh simpul dari sisi-sisi yang baru saja dipilih dimasukkan ke dalam SV.
Ulangi tahap terakhir sampai SV kosong.	

Setelah terbentuk graf skeletal bangun, yang terakhir dilakukan adalah pemilihan sisi untuk dijadikan *seam*. Kita mulai dengan membandingkan sebuah simpul V yang ada di graf keterhubungan bangun dengan simpul yang berkorespondensi di graf skeletal. Sisi yang ada di graf keterhubungan dan bersisian dengan V dan V_t namun tidak ada di graf skeletal menandakan kita harus menandakan sisi yang membatasi muka yang berkorespondensi dengan V (yaitu muka yang titik tengahnya berkorespondensi dengan V) dengan muka yang berkorespondensi dengan V_t. Proses ini diterapkan pada seluruh simpul yang ada di kedua graf (jumlah simpul dalam kedua graf selalu sama).

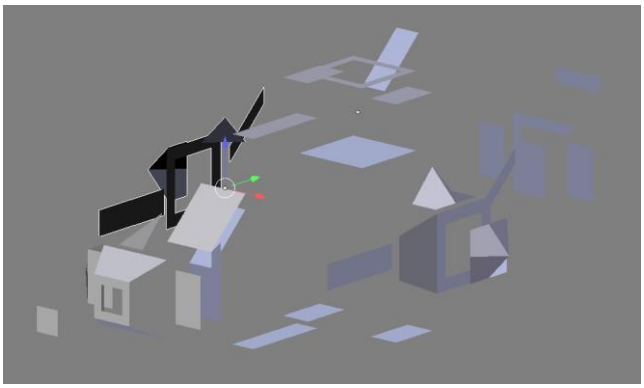


Gambar 7. Hasil dari algoritma alternatif II.

D. Perbandingan dengan Algoritma Lain

Pada bagian ini penulis akan mendeskripsikan algoritma lain yang banyak dipakai untuk *UV-Unwrapping* secara singkat karena detilnya berada di luar lingkup makalah ini. Selain itu, penulis juga akan membandingkan algoritma yang penulis bahas dalam makalah ini (selain algoritma naif) yang digunakan untuk penentuan *seams* dalam *UV-Unwrapping*.

Pada dasarnya, algoritma yang digunakan adalah memproyeksikan topologi permukaan ke berbagai bidang. Umumnya, topologi permukaan model secara kasar diproyeksikan ke bidang depan, bidang kanan, bidang kiri, bidang atas, bidang bawah, dan bidang belakang (urutannya tergantung aplikasi yang mengimplementasikannya). Selain itu terdapat pula proyeksi secara *spherical* dan *cylindrical*. Masing-masing menggunakan bola / tabung yang melingkupi model yang akan di *UV-Unwrapping* lalu topologi model tersebut diproyeksikan ke bola / tabung tersebut.



Gambar 8. Ilustrasi kasar metode proyeksi dari model di gambar 2.

UV Map hasil dari algoritma selalu planar, karena bangun ruang 3D tidak lain adalah kombinasi dari bangun planar 2D. Hasil dari algoritma ini juga cukup baik, namun dalam banyak kasus, pembentukan *island* sering tidak dapat dihindarkan dan terkadang membingungkan pemodel karena bingung muka yang ada di *UV Map* berkorespondensi dengan muka yang mana di model yang ia miliki. Sisi positifnya adalah pemodel dapat mengatur sebagian *island* yang ada di *UV Map*

Algoritma alternatif II yang penulis bahas sebelumnya memang selalu menghasilkan *UV Map* dengan jumlah *island* yang di kebanyakan kasus tidak ada. Memang, hasil ini mungkin akan menghasilkan orientasi gambar yang berbeda-beda ketika diproyeksikan ke model dan terkadang ada masalah penyambungan (*stitching*) yang kadang kurang enak dilihat. Di luar hal itu, tampak belum ada masalah sejauh yang penulis coba. Untuk implementasi manual, algoritma alternatif II memang lebih rumit dipraktikkan daripada algoitma alternatif I, namun algoritma alternatif II menghasilkan *UV Map* yang jauh lebih baik daripada algoritma alternatif I untuk model dengan topologi yang rumit.

Algoritma alternatif II yang penulis bahas sebelumnya dapat disarikan menjadi tiga tahapan utama yaitu:

- Buat graf keterhubungannya.
- Buat graf skeletal dari graf keterhubungan yang terlah dibuat.
- Interpretasi graf skeletal untuk pemilihan *seams*.

Algoritma untuk tahapan pertama bila diimplementasikan dengan baik dan benar dapat berjalan dalam $O(n)$ dengan n menyatakan jumlah muka yang terdapat dalam model. Algoritma untuk tahapan kedua bila diimplementasikan dengan baik dan benar juga dapat berjalan dalam $O(m)$ dengan m menyatakan jumlah sisi yang terdapat dalam graf keterhubungan. Untuk tahapan ketiga seharusnya dapat dijalankan juga dalam $O(p)$ dengan p menyatakan banyaknya seluruh sisi yang terdapat dalam model. Karena $n \leq m \leq p$, maka algoritma ini berjalan optimal dalam $O(p)$. Namun, supaya algoritma ini berjalan optimal, harus didukung struktur data yang merepresentasikan struktur *mesh* model yang tepat, karena jika struktur data ternyata tidak mendukung algoritma I untuk berjalan optimal (misalnya pada saat menemukan adanya muka yang memiliki sisi persekutuan), maka bisa saja kompleksitas algoritma membengkak menjadi faktorial atau eksponensial, yang sangat tidak diharapkan.

IV. Pekerjaan yang Akan Datang

Algoritma-algoritma yang penulis bahas masih berupa rancangan. Pengujian algoritma masih dilakukan secara manual, sehingga model yang diuji pun tidak memiliki banyak muka dan sisi. Algoritma-algoritma ini butuh disempurnakan lebih jauh dengan mengimplementasikannya ke dalam program yang nyata, sehingga dapat dilakukan pengujian ke model yang memiliki muka lebih banyak dan topologi lebih rumit. Algoritma-algoritma ini juga butuh perbaikan supaya dapat menyesuaikan dengan struktur yang dipakai secara general oleh paket aplikasi grafis 3D. Selain itu, algoritma alternatif II memiliki beberapa masalah di muka dengan jumlah sisi > 4 dan walaupun proporsi gambar di seluruh model tampak normal, orientasi keseluruhan cenderung kurang sempurna.

V. Kesimpulan

Proses *UV-Unwrapping* merupakan proses yang krusial dalam tahap pemodelan dalam grafis 3D, baik animasi, pembuatan boneka untuk 3D *printing*, maupun pemaketan model. Terkadang, pemodel tidak puas dengan hasil yang dibentuk oleh algoritma automisasi *UV-Unwrapping* dengan metode proyeksi yang penulis gambarkan secara singkat pada bagian III D, sehingga mereka memilih untuk melakukan *UV-Unwrapping* secara manual. Oleh karena itu, sering sekali pemodel dipusingkan dengan pemilihan *seams* dalam model yang mereka buat supaya tercipta hasil yang baik, terlebih pemodel pemula.

Algoritma-algoritma yang penulis paparkan di dalam makalah ini masing-masing memiliki kelebihan dan kekurangan. Algoritma naif meskipun mudah dimengerti, namun algoritma ini tidak bisa dipraktikkan karena prosesnya yang sangat lama baik untuk diterapkan manual maupun dengan program. Oleh karena itu, penulis memberikan dua alternatif. Alternatif I cocok untuk pemodel yang ingin *UV-Unwrapping* secara cepat, namun hasilnya terkadang buruk. Alternatif II lebih tepat untuk diaplikasikan, meskipun lebih rumit untuk diterapkan manual. Algoritma ini bila diimplementasikan secara baik dan benar, maka algoritma ini berjalan dalam waktu polinomial. Namun, dibutuhkan penyempurnaan dan penelitian lebih lanjut terhadap algoritma-algoritma ini supaya dapat benar-benar diimplementasikan dalam praktik sehari-hari.

UCAPAN TERIMA KASIH

Puji syukur penulis panjatkan kepada Allah SWT karena berkat limpahan nikmatnya penulis bisa menyelesaikan makalah ini tepat waktu. Penulis juga mengucapkan terima kasih kepada Bapak Dr. Ir. Rinaldi Munir dan Ibu Dr. Nur Ulfa Maulidevi, S.T., M.T. yang merupakan kedua dosen penulis untuk mata kuliah ini atas kesediannya mengajarkan ilmunya kepada penulis. Penulis juga mengucapkan terima kasih atas seluruh pihak yang membantu penulis baik secara langsung maupun tidak langsung.

REFERENSI

[1] Munir, Rinaldi. 2007. "*Diktat Kuliah IF2211: Strategi Algoritma"*. Bandung : Institut Teknologi Bandung.

[2] Rosen, Kenneth H. 2012. "*Discrete Mathematic and Its Applications, Seventh Edition"*. New York : McGrawHill

[3] pellacini.di.uniroma1.it/teaching/graphics13a/slides/13_textures.pdf . Diakses pada tanggal 8 Mei 2016.

PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 8 Mei 2016

Muhammad Reifiza / 13514103