Penerapan Strategi Algoritma Menyelesaikan Kompleksitas Pixel Maze secara Otomatis pada Virupizxel

Biolardi Yoshogi / 13509035
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
vsio@*students.itb.ac.id*

***Absrkt*—Pixel maze adalah maze yang dibuat dengan pixel melalui tahap pixel art. Pixel maze yang ada di Virupizxel adalah buatan saya sendiri (kecuali fanart[hanya bagian gambar, bukan pixel mazenya]). Pixel maze ini terdiri dari gabungan beberapa warna seperti gambar buah, tetapi ada path maze di dalamnya sehingga menentukan pathnya tidak akan semudah pixel maze biasa. Oleh karena itu, diterapkan Strategi Algoritma berupa agar bisa menentukan mana yang path dan mana yang wall di dalam pixel maze tersebut.**

***Kaca Kunci*—pixel maze, algoritma *divide and conquer*, solusi, kompleks**

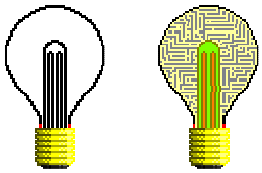
I. PENDAHULUAN

Pixel maze adalah maze yang dibuat dengan pixel melalui tahap pixel art. Pixel maze buatan saya yang ada di blog saya Virupizxel (www. virupizxel.blogspot.com/) ini mulai dari yang sederhana hingga yang kompleks berdasarkan banyaknya warna, shading, frame animasi, besar ukuran, dan hal lain yang mungkin ditambahkan ke depannya.

Pixel maze yang saya buat dan berada di Virupizxel itu juga memiliki nama berdasarkan tanggal yang mana duadigit hari berdasarkan potongan kata angka dari bahasa Indonesia, dua huruf dari depan untuk puluhan dan tiga huruf dari depan untuk satuan, bulan berdasarkan tiga potongan huruf dari bahasa Inggris, dan empat digit tahun berdasarkan rumus penamaan pada tabel periodik pada kimia. Contohnya disertai gambar berdasarkan yang sudah diposting:

Gambar 1.1 Binilununaugsanol

Gambar **1.**2 Binilununocttisat



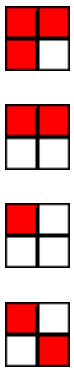
Gambar **1.**3 Binilunundecnoena: a. frame 1; b.frame 2

Pada Gambar **1.**1 (pixel maze pertama yang diposting di blog Virupizxel), Binilununaugsanol menunjukkan “Bi-nil-un-un-” merupakan tahun 2011, “-aug-“ merupakan bulan Agustus, “-sa-nol” merupakan hari satu pada puluhan dan nol pada satuan. Maze ini merupakan maze yang sederhana.

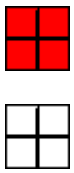
Pada Gambar **1.**2, Binilununocttisat menunjukkan “Bi-nil-un-un-” merupakan tahun 2011, “-oct” merupakan bulan Oktober, “-ti-sat” merupakan hari 3 pada puluhan dan satu pada satuan. Maze ini mulai lebih kompleks karena melibatkan warna lebih dari yang dimiliki maze pada gambar 1 dan mulai menyerupai suatu bentuk yaitu sebuah labu.

Pada Gambar **1.**3, Binilunundecnoena menunjukkan “Bi-nil-un-un-” merupakan tahun 2011, “-dec” merupakan bulan Desember, “-no-ena” merupakan hari 0 pada puluhan dan enam pada satuan. Maze ini mulai lebih kompleks juga karena diimplementasikannya animasi pada pixel maze ini.

Pixel maze di dalam blog saya juga menerapkan peraturan khusus bahwa path maze dalam ruang terpisah 4 pixel hanya boleh diisi 3 pixel path atau wall. Ilustrasi seperti di bawah ini (dengan asumsi bahwa 1 kotak sama dengan 1 pixel, yang path berwarna putih dan wall berwarna merah:



Gambar **1.**4 Yang diperbolehkan



Gambar **1.**5 Yang tidak diperbolehkan

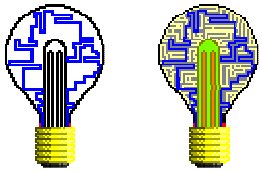
Penyelesaian pixel maze ini masih secara manual. Jika pixel maze yang diselesaikan itu kecil, hal ini masih tidak masalah dan hanya menkonsumsi waktu relatif tidak lama. Akan tetapi, jika ukurannya mencapai ratusan atau ribuan (pixel maze terbesar yang pernah saya buat berukuran 501x5001 [bisa diperiksa di tautan ini: http://virupizxel.blogspot.com/2011/08/binilununmingtiemp.html] dan pembuatannya memakan waktu berjam-jam karena pembuatannya juga secara manual). Ini masih pixel sederhana. Belum pixel kompleks yang terbesar pernah saya buat hingga ukuran kotornya mencapai 5555x5555 pixels (bisa diperiksa di tautan ini: http://virupizxel.blogspot.com/2011/09/binilununmingtilim.html) Ditambah lagi, jika diselesaikan secara manual juga akan memakan waktu dan akan ada beberapa faktor penghalang lainnya seperti :

- Tersesat sehingga harus mundur kembali dan tersesat lagi
- Efek pusing
- Sulit konsentrasi

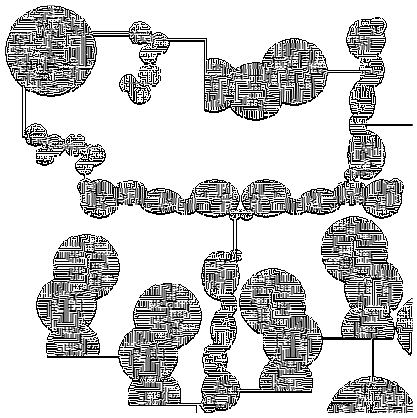
Oleh karena itu, perlu adanya algoritma yang bisa menyelesaikan pixel maze secara otomatis. User tinggal memasukkan file gambar yang berisi pixel maze dan hasilnya akan keluar file gambar dengan garis yang menunjukkan ke arah solusi ditambah lagi solusi yang terdekat dari titik mulai hingga titik akhir.

Gambar 1.6 Solusi Binilununaugsanol

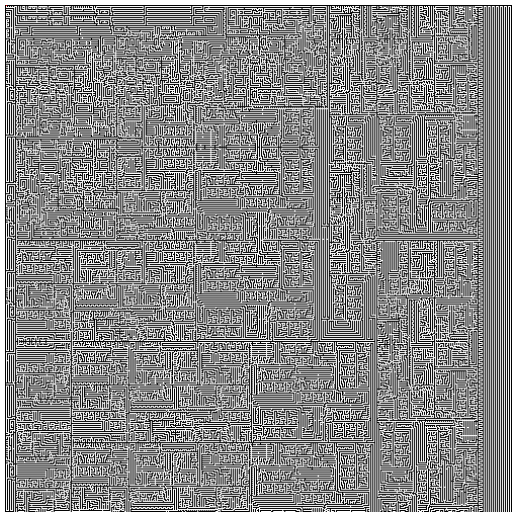
Gambar **1.**7 Solusi Binilununocttisat



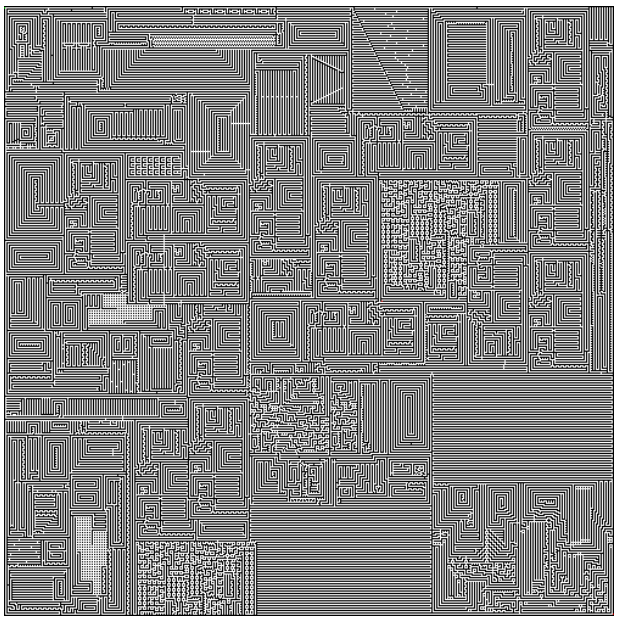
Gambar **1.**8 Solusi Binilunundecnoena: a. frame 1; b.frame 2



Gambar 1.9 Pixel maze (sebagian) berukuran 5555x5555 pixels yang ukurannya diperkecil dan dipotong



Gambar 2.0 Pixel maze (sebagian) berukuran 501x5001 pixels yang ukurannya diperkecil dan dipotong



Gambar 2.1 Pixel maze (sebagian) berukuran 501x501 pixels yang ukurannya diperkecil dan dipotong

Pada gambar di atas (kecuali gambar 1.9, 2.0, dan 2.1), solusi tersebut masih belum yang terpendek dari titik awal hingga titik akhir. Hal ini juga mendorong diperlukankan adanya solusi yang bisa menemukan jarak terdekat dari titik awal hingga akhir.

Pada gambar di atas (khusus Gambar 1.9 hingga 2.1), terlihat bahwa pixel maze tersebut relatif cukup besar dan penyelesaian bisa memakan waktu yang lama ditambah lagi waktu psikir pemain yang kemungkinan akan terganggu karena ukurannya yang cukup besar (dengan asumsi pixel maze tersebut dengan semua bagiannya dan tidak di zoom-out).

Permasalahan sekarang adalah bagaimana menentukan mana yang jalur pixel maze dan mana yang bukan. Pada Gambar 2 dan Gambar 3, terlihat banyak kombinasi warna pada path dan wall sehingga akan mempersulit menentukan mana yang path dan mana yang wall jika ingin diselesaikan secara otomatis. Dalam makalah ini, saya menggunakan pendekatan Algoritma Brute Force, Algoritma Runut-Balik, dan *Algoritma Divide and Conquer*.

II. TEORI DASAR

Sebelum memulai solusi strategi algoritma yang diterapkan pada pixel maze kompleks ini, akan dijelaskan beberapa teori algoritma yang akan digunakan pada kasus ini.

II. 1. Algoritma Brute Force

- Brute force: pendekatan yang lempang (straightforward) untuk memecahkan suatu masalah.
- Biasanya didasarkan pada:
  – pernyataan masalah (problem statement)
  – definisi konsep yang dilibatkan.
- Algoritma brute force memecahkan masalah dengan:
  – sangat sederhana,
  – langsung,
  – jelas (obvious way)

- Contoh-contoh :
  - Mencari elemen terbesar (terkecil)
  - Pencarian beruntun (Sequential Search)
  - Menghitung an (a> 0, nadalah bilangan bulat tak-negatif)
  - Menghitung n! (nbilangan bulat tak-negatif)
  - Mengalikan dua buah matriks, Adan B
  - Menemukan semua faktor dari bilangan bulat n (selain dari 1 dan n itu sendiri)
  - Uji keprimaan
  - Beberapa metode pengurutan memakai algoritma ini seperti Bubble Sort dan Selection Sort

- Karakteristik :
  - Algoritma brute force umumnya tidak “cerdas” dan tidak mangkus, karena ia membutuhkan jumlah langkah yang besar dalam penyelesaiannya
  - Algoritma brute force lebih cocok untuk masalah yang berukuran kecil
  - Meskipun bukan metode yang mangkus, hampir semua masalah dapat diselesaikan dengan algoritma brute force

II.2 Algoritma Runut-balik

- Runut-balik (backtracking) adalah algoritma yang berbasis pada DFS untuk mencari solusi persoalan secara lebih mangkus.
- Runut-balik, yang merupakan perbaikan dari algoritma brute-force, secara sistematis mencari solusi persoalan di antara semua kemungkinan solusi yang ada.
- Memangkas (pruning) simpul-simpul yang tidak mengarah ke solusi
- Algoritma runut-balik banyak diterapkan untuk program games:
  - permainan tic-tac-toe
  - menemukan jalan keluar dalam sebuah labirin
  - catur

- Properti Umum Metode Runut-balik :
  - Solusi persoalan
  - Fungsi pembangkit
  - Fungsi pembatas

III. 3Algoritma *Divide and conquer*

- Divide: membagi masalah menjadi beberapa upa-masalah yang memiliki kemiripan dengan masalah semula namun berukuran lebih kecil (idealnya berukuran hampir sama)
- Conquer: memecahkan (menyelesaikan) masing-masing upa-masalah (secara rekursif)
- Combine: mengabungkan solusi masing-masing upa-masalah sehingga membentuk solusi masalah semula
- Contoh-contoh masalah :
  - Mencari nilai minimum dan maksimum (MinMaks)
  - Mencari pasangan titik yang jaraknya terdekat (Closest Pair)
  - Beberapa metode pengurutan memakai algoritma ini seperti Quick Sort dan Selection Sort

III. METODE PEMECAHAN MASALAH

Pertama kali buat asumsi berikut ini:

- Masukan adalah berupa file image pixel (.gif, .png)
- Yand dimasukkan selalu berupa pixel maze

Pada kasus ini, digunakan tiga strategi algoritma (Brute Force, Runut-Balik, dan *Divide and conquer*) untuk memecahkan masalah pada kasus pixel maze ini.

Struktur data yang digunakan adalah array dua dimensi. Satu dimensi mempresentasikan posisi horizontal pixel maze dan satu dimensi lagi mempresentasikan posisi vertikal pixel maze. Struktur data ini menyimpan informasi berupa apakah pixel ini merupakan wall atau path.

Berikut ini fungsi-fungsi yang akan digunakan dalam pemecahannya :

procedure PixelMazeSolution
➔ Menentukan path dan mencari solusi terdekat pada pixel maze serta menghasil file .png. Fungsi ni merupakan driver.

function ScanPixelColor(input matrix_: Matrix) → List
➔ Mengembalikan list frekunsi kemunculan warna.

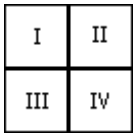
procedure ScanPixelMaze(input/output matrix_: Matrix)
➔ Memeriksa apakah maze merupakan path atau wall berdasarkan frekuensi warna dan disekitarnya. Asumsi jika hanya terdapat 2 buah pada satu warna, dianggap sebagai start dan finish-nya. Jika indeks vertical atau horizontalnya bernilai nol tapi tidak keduanya, diasumsikan langsung sebagai wall.

procedure SolveMaze(input/output matrix_: Matrix
➔ Mencari solusi terdekat pada sebuah pixel maze dan menyimpannya sebagai .png

Berikut ini pemecahan-pemecahan dengan ketiga algoritma tersebut:

- Algoritma Divide Conquer

Dengan algoritma ini, area yang diperiksa akan dibagi menjadi empat bagian seperti gambar di bawah ini:



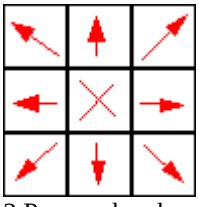
Gambar 3.1 Pembagian daerah menjadi 4 bagian

Setelah itu, daerah kecil-kecil tadi akan dibagi lagi hingga tersisa menjadi 1 pixel. Algoritma ini memperkecil jumlah pengulangan yang digunakan. Dalam pembagian ini, dilakukan juga ScanPixelColor untuk menghitung banyaknya kemunculan warna.

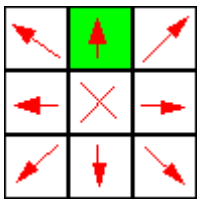
Strategi algoritma ini juga digunakan untuk kasus pixel maze yang telah disebutkan tadi di pendahuluan yang ukurannya mencapai 501 x 5001 pixels yang mana jika menggunakan algoritma brute force pada fungsi yang telah disebutkan tadi, akan memakan waktu lebih lama lagi.

- Algoritma Brute Force

Algoritma ini diimplementasikan pada ScanPixelMaze. Algoritma mengecek ulang pixel maze dengan frekuensi warna yang didapat sebelumnya. Prosedur ini mengecek juga yang di sekitarnya.

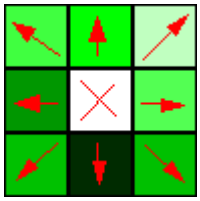

Gambar 3.2 Pengecekan ke semua arah

Berdasarkan gambar tadi, titik X memeriksa semua arah yang diperiksanya sebelum menentukan apakah warna tersebut sebaiknya menjadi path, wall, atau goal. Ditambah lagi, jika di sekitarnya adalah pixel transparan, maka x akan langsung ditentukan sebagai wall (kecuali untuk titik awal dan titik akhir).



Gambar 3.3 Pengecekan ke semua arah lagi dengan asumsi warna hijau adalah transparan

Pada kasus Gambar 3.3 dengan asumsi warna hijau adalah transparan, maka x akan bernilai wall karena berdasarkan syarat yang telah dibuat sebelumnya yang membuat x tersebut menjadi wall.



Gambar 3.4 Pengecekan ke semua arah lagi dengan asumsi warna hijau adalah memang warna, dan dengan gradasi warna yang berbeda yang harus dihadapi dan diperiksa

Pada kasus gambar 3.4, dilakukan pengecekan pada gambar dengan asumsi hijau adalah warna. Meskipun ada kedekatan dengan warna, karena adanya perhitungan jumlah warna dengan dan masing-masing diambil berdasarkan nilai hexnya, tidak akan masalah kecuali jika diambil pendekatan subjektif yang akan mempengarauhi akurasi

Karena adanya pengecekan ulang secara brute force pada pixel maze ini yang mana mengecek satu persatu perpixel, maka kira-kira akan memakan waktu relatif lebih lama terutama pixel maze yang berukuran ratusan hingga ribuan yang telah disebutkan di pendahuluan sebelumnya

- Algorita Runut Balik:

Ketika semua path dan wall sudah ditentukan oleh algoritma tadi dengan sistem indeks apakah wall atau bukan, maka bisa digunakan prosedur SolveMaze (sama dengan cara pemecahan maze umumnya). Meskipun pixel maze tersebut kompleks, karena pengecek berdasarkan indek dari hasil fungsi dan prosedur tadi, penyelesaian pixel maze dapat dilakukan secara biasa yang mana biasanya dilakukan algoritma runut-balik pada maze pada umumnya.

IV. ANALISIS

Dengan implementasi algoritma brute force, runut-balik, dan *divide and conque*, penyelesaian secara otomatis pada pixel maze dapat dilakukan. Akan tetapi, tentu saja tetap akan ada kesalahan terutama pada pemeriksaan warna karena bisa saja warna yang diproses akan banyak perbedaan yang terutama disebabkan oleh efek shading pada pixel maze tersebut. Ditambah lagi, diterapkannya algoritma Brute Force pada suatu kasus yang besar (pixel maze berukuran ratusan hingga ribuan pixel), akan kemungkinan tetap akan memakan waktu lebih banyak karena terjadinya pengulangan dari ratusan hingga ribuan kali ditambah lagi harus memeriksa di sekitarnya yang berarti aa delapan pengulangan yang mana hal ini menambah jumlah pengulangan secara drastis.

V. KESIMPULAN

Dari metode dan analisis yang didapat, dapat disimpulkan bahwa algoritma Brute Force, Runut-Balik, dan *Divide and Conquer*, dapat digunakan untuk memecahkan masalah pixel maze kompleks yang terdiri dari berbagai macam warna dan frame animasi. Meskipun tidak mungkin sempurna dan tentu saja akan ada kesalahan karena masih perlunya pendekatan subjektif, metode ini masih memungkinkan untuk mendapatkan hasil pixel maze secara otomatis. Harapan dari saya adalah agar ada algoritma atau solusi lain yang lebih efektif sehingga mampu menyelesaikan segala macam kasus pixel maze yang terdapat pada blog Virupizxel tersebut.

REFERENSI

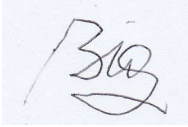
[1] Munir, Rinaldi. “Diktat Kuliah IF3051 Strategi Algoritma”. Institut Teknologi Bandung. 200

[2] http://virupizxel.blogspot.com. Waktu akses : 9 Desember 2011, jam 7.00.

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, 09 Desember 2011



Biolardi Yoshogi / 13509035