

Simulasi Pergerakan Evakuasi Bencana Tsunami Menggunakan Algoritma Boids dan Pathfinding

Movement of the Tsunami Evacuation Simulation Using Boids and Pathfinding Algorithm

I Made Pasek Mudhana¹, Mauridhi Hery Purnomo², Supeno Mardi Susiki Nugroho³

*Jurusan Teknik Elektro, Institut Teknologi Sepuluh Nopember Surabaya^{1,2,3}
made13@mhs.ee.its.ac.id¹*

Abstrak

Musibah bencana alam tsunami yang terjadi menerjang Indonesia khususnya Aceh dan sekitarnya yang memakan banyak korban, telah memberikan gambaran kebutuhan evakuasi dini pada saat terjadi suatu musibah, khususnya Tsunami. Penelitian ini mensimulasikan pergerakan kerumunan orang untuk bergerak menuju suatu titik evakuasi pada saat terjadinya gempa bumi yang diperkirakan menimbulkan bahaya tsunami. Simulasi ini menentukan jarak terdekat/terpendek dari posisi individu, meminimalkan terjadinya tabrakan dalam menghindari segala hambatan yang ditemui baik hambatan statis maupun dinamis yang ditemui pada saat melewati rute jalan yang telah ditentukan. Penerapan model kerumunan diselesaikan dengan menggunakan algoritma *boids*, yang didalamnya terdiri dari algoritma *flocking*, *obstacle avoidance*, *collision detection*, dengan ditambahkan dengan algoritma *pathfinding*.

Kata Kunci: Simulasi, Evakuasi, *Boids*, *Flocking*, *Obstacle Avoidance*, *Collision Detection*, *Pathfinding*.

1. Pendahuluan

Proses evakuasi adalah proses yang sangat membutuhkan ketepatan dalam pengalokasian waktu, terutama didalam penanganan kejadian bencana yang bersifat kompleks seperti evakuasi tsunami. Seperti diketahui bersama tsunami adalah terjadinya gelombang besar yang disebabkan oleh gempa bumi yang berpusat dibawah laut. Biasanya gelombang besar tsunami akan menghantam daerah pesisir pantai dengan kekuatan yang cukup dahsyat yang bisa mengakibatkan korban jiwa dan harta benda.

Dalam kondisi seperti ini, perlu dilakukan evakuasi yang bertujuan untuk menyelamatkan penduduk dari bahaya tsunami, dari tempat yang berpotensi terkena tsunami ketempat yang dianggap aman. Dengan evakuasi diharapkan dapat mengurangi atau meminimaliasai jumlah korban jiwa. Perihal yang paling erat hubungannya dengan evakuasi adalah waktu, semakin lama proses evakuasi atau semakin besar waktu evakuasi yang dibutuhkan maka akan semakin banyak jiwa yang terancam. Selain itu proses evakuasi juga dipengaruhi oleh banyaknya penduduk yang harus diselamatkan serta lokasi dimana para penduduk tersebut berada. Jadi semakin banyak jumlah penduduk dan semakin kompleks bentuk denah pemukiman maka proses evakuasi akan membutuhkan waktu yang lebih lama.

Dengan mengacu pada penelitian sebelumnya yang berjudul Simulasi Pergerakan

Pengunjung Mall Menggunakan Flocking dan Obstacle Avoidance (Dewi, 2012) tentang kelebihan algoritma *boids*, pada penelitian ini akan menerapkan algoritma *boids* dengan menggunakan *flocking*, *obstacle avoidance*, *collision detection* dengan *pathfinding* untuk mensimulasikan pergerakan orang pada saat evakuasi bencana tsunami menuju suatu titik evakuasi dalam menghindari hambatan statis berupa benda diam di dalam perjalanan menuju titik evakuasi dan hambatan dinamis berupa benda bidang bergerak dan meminimalkan frekuensi jumlah tabrakan yang terjadi antar penduduk/orang karena meningkatnya jumlah populasi.

1.1 Dasar Teori

1.1.1 Definisi Bencana dan Tsunami

Undang-undang Nomor 24 Tahun 2007 Tentang Penanggulangan Bencana menyebutkan definisi bencana sebagai berikut:

Bencana adalah peristiwa atau rangkaian peristiwa yang mengancam dan mengganggu kehidupan dan penghidupan masyarakat yang disebabkan, baik oleh faktor alam dan/atau faktor nonalam maupun faktor manusia sehingga mengakibatkan timbulnya korban jiwa manusia, kerusakan lingkungan, kerugian harta benda, dan dampak psikologis.

Definisi tersebut menyebutkan bahwa bencana disebabkan oleh faktor alam, non alam, dan manusia. Oleh karena itu, Undang-Undang Nomor 24 Tahun 2007 tersebut juga mendefinisikan mengenai bencana alam, bencana nonalam, dan bencana sosial.

Tsunami berasal dari bahasa Jepang yang berarti gelombang ombak lautan (*tsu* berarti lautan, *nami* berarti gelombang ombak). Tsunami adalah serangkaian gelombang ombak laut raksasa yang timbul karena adanya pergeseran di dasar laut akibat gempa bumi.

1.1.2 Definisi Evakuasi

Evakuasi adalah perpindahan langsung dan cepat dari orang-orang yang menjauh dari ancaman atau kejadian yang sebenarnya dari bahaya. Contoh berkisar dari evakuasi skala kecil sebuah bangunan karena ancaman bom atau kebakaran sampai pada evakuasi skala besar sebuah distrik karena banjir, penembakan atau mendekati badai. Dalam situasi yang melibatkan bahan-bahan berbahaya atau kontaminasi, pengungsi sebaiknya didekontaminasi sebelum diangkut keluar dari daerah yang terkontaminasi.

Rencana evakuasi darurat dikembangkan untuk memastikan waktu evakuasi teraman dan paling efisien bagi semua penduduk yang diharapkan dari suatu bangunan, kota, atau wilayah. Sebuah tolok ukur kinerja (benchmark) "waktu evakuasi" untuk bahaya yang berbeda dan kondisi dibuat. *Benchmark* ini dapat dilakukan melalui penggunaan praktik terbaik, peraturan atau menggunakan simulasi, seperti model aliran manusia dalam sebuah bangunan, untuk menentukan *benchmark*. Perencanaan yang tepat akan menggunakan beberapa jalan keluar serta teknologi untuk memastikan evakuasi penuh dan lengkap. Pertimbangan untuk sejumlah situasi pribadi yang mungkin mempengaruhi kemampuan individu melakukan evakuasi. Situasi-situasi pribadi itu mungkin termasuk sinyal alarm yang menggunakan tanda atau sinyal yang bisa didengar dan dilihat.

Peraturan-peraturan seperti kode bangunan dapat digunakan untuk mengurangi kemungkinan panik dengan memungkinkan individu menyiapkan kebutuhan untuk mengevakuasi diri tanpa menyebabkan alarm. Perencanaan yang tepat akan menerapkan pendekatan semua-bahaya sehingga rencana itu dapat digunakan kembali untuk beberapa bahaya yang mungkin ada. (Abraham, 1994).

1.1.3 Perilaku Orang Berjalan

Dalam kehidupan nyata, kecepatan berjalan setiap orang tidaklah sama, tergantung oleh banyak faktor, antara lain: umur, jenis kelamin, waktu berjalan (pagi, siang atau malam), tujuan perjalanan, reaksi terhadap environment sekitar, temperatur udara dan lain-lain. Beberapa pakar transportasi menggunakan kecepatan rata-rata 1,20 m/detik (72 m/menit), namun untuk pejalan kaki yang cenderung berjalan lebih lambat, menggunakan kecepatan 0,90 s/d 1,00m/detik (54-60m/menit)(Aspelin, 2005).

1.1.4 Algoritma Boids

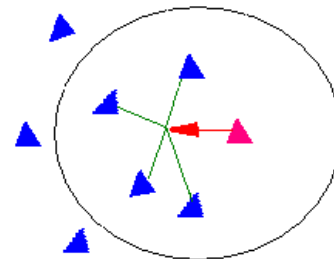
Boids adalah sebuah algoritma yang merepresentasikan gerak dari sebuah kawanan. Perilaku yang dihasilkan sangat mirip dengan kumpulan ikan atau kawanan burung. Gerak *boids* dihasilkan dari tiga aturan sederhana yaitu *cohesion*, *alignment*, *separation* (Reynolds, 2010).

1.1.4.1 Separation

Pembatasan jika sebuah agen terlalu dekat dengan agen lainnya, dengan cara melakukan penyesuaian arah dan kecepatan untuk menghindari benturan (*collision*).

Agen akan menjaga jarak agar tidak nempel dengan flocksmate atau tetang ganya. Aturan ini mengarahkan (*steering*) agar *Boids* bergerak menghindari kondisi yang padat (*crowded*) oleh kawanan tetangganya. Hal ini memungkinkan *Boids*:

1. Menghindari terjadinya tabrakan
2. Menjaga agar *boids* tetap terpisah pada jarak pisah tertentu yang realistis atau tidak terlalu berdekatan



Gambar 1. Separation (Reynold, 2010)

Separation dapat di rumuskan sesuai pada persamaan berikut;, (Cui, 2006).

$$d(P_x, P_b) \leq d_2 \Rightarrow \vec{v}_{sr} = \sum_{i=1}^n \frac{\vec{v}_x + \vec{v}_b}{d(P_x, P_b)}$$

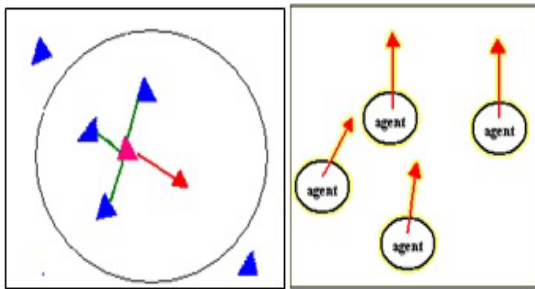
Dimana V_{sr} adalah kecepatan yang ditentukan oleh aturan *separation* seperti pada gambar 1, $d(P_x, P_b)$ adalah jarak antara *boid* x dengan tetangga b, V_x dan V_b adalah kecepatan *boid* x dan *boid* b, d_2 adalah nilai jarak yang telah ditetapkan.

1.1.4.2 Alignment

Mengambil rata-rata dari semua percepatan agen yang lain dan melakukan penyesuaian percepatan untuk pindah kearah kelompok. Mengarahkan agent menjuposisi rata-rata tetangga. Aturan ini mengarahkan (*steering*) agar *Boids* bergerak ke arah yang merupakan tujuan dari sebagian besar kawanan di tetangga lokalnya.

Boids berusaha untuk menyesuaikan kecepatannya (arah, kecepatan bergerak) dengan kecepatan tetangga-tetangganya. Hal ini memungkinkan *Boids*:

1. Mengimbangi pemisahan
2. Membuat *boids* bergerak pada satu arah tujuan umum yang sama



Gambar 2. Alignment (Reynold, 2010)

Alignment dapat di rumuskan sesuai pada persamaan berikut, (Cui, 2006).

Dimana V_{ar} adalah kecepatan yang ditentukan

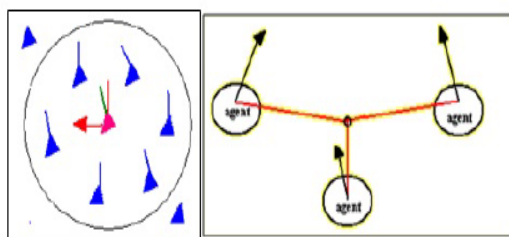
$$d(P_x, P_b) \leq d_1 \cap d(P_x, P_b) \geq d_2 \Rightarrow \vec{v}_a = \frac{1}{n} \sum_i \vec{v}_i$$

oleh aturan alignment seperti pada gambar 2, $d(P_x, P_b)$ adalah jarak antara *boid* x dengan tetangga b, n adalah total jumlah tetangga, V_x adalah kecepatan *boid* x, d_1 dan d_2 adalah nilai jarak yang telah ditetapkan.

1.1.4.3 Cohesion

Menghitung pusat keseluruhan kelompok dan mengarahkan agen ke arah titik pusatnya. Agen akan mencoba untuk tetap dekat dengan kelompoknya. Aturan ini mengarahkan (*steering*) agar *Boids* (agen) bergerak maju ke arah yang merupakan tujuan dan sebagian besar kawanannya di tetangga lokalnya. Hal ini memungkinkan *Boids*:

1. Tetap bersama-sama dengan kawanannya lokalnya
2. Melakukan kegiatan pengumpulan beberapa kawanannya maupun pemisahan kawanannya ke dalam 2 kelompok



Gambar 3. Cohesion (Reynold, 2010)

Menghitung pusat keseluruhan kelompok dan mengarahkan agent ke arah titik pusatnya. Yakni, jumlah posisi dari semua tetangga dibagi dengan jumlah tetangga. Akhirnya, menggunakan perilaku seek agen bergerak menuju titik pusat.

Cohesion dapat di rumuskan sesuai pada persamaan berikut, (Cui, 2006).

$$d(P_x, P_b) \leq d_1 \cap d(P_x, P_b) \geq d_2 \Rightarrow \vec{v}_a = \sum_i (P_x - P_b)$$

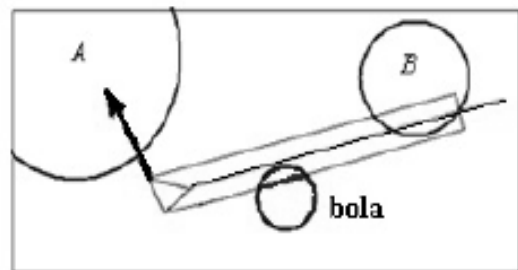
Dimana V_{cr} adalah kecepatan yang ditentukan oleh aturan *cohesion* seperti pada gambar 3, $d(P_x, P_b)$ adalah jarak antara *boid* x dengan tetangga b, d_2 adalah nilai jarak yang telah ditetapkan, sedangkan $(P_x - P_b)$ adalah perhitungan arah *vector point*.

1.2 Obstacle Avoidance

Perilaku menghindari hambatan (*obstacle avoidance*) memberikan kemampuan karakter untuk manuver di *environment* dengan menghindari hambatan sekitarnya.

Ada perbedaan penting antara menghindari hambatan (*obstacle avoidance*) dan perilaku melarikan diri (*flee*). *Flee* akan selalu mengarahkan karakter untuk menjauh dari lokasi tertentu, sedangkan *obstacle avoidance* tindakan akan diambil hanya jika suatu hambatan yang terdekat terletak tepat di depan karakter. Sebagai contoh, jika sebuah mobil mengemudi sejajar dengan dinding, *obstacle avoidance* akan mengambil tindakan korektif kemudi, tapi *flee* akan berusaha untuk berpaling dari dinding, akhirnya mengemudi tegak lurus dengan dinding. Implementasi dari perilaku *obstacle avoidance* berhubungan dengan penghindaran rintangan dimana tidak harus terjadi tabrakan. Bayangkan sebuah pesawat berusaha untuk menghindari gunung.

Tujuan dari perilaku *obstacle avoidance* adalah untuk menjaga sebuah silinder imajiner (sebagai hambatan) yang berada di depan karakter bola, seperti yang diilustrasikan pada Gambar di bawah. Silinder A dan B terletak disepanjang sumbu didepan karakter bola. Perilaku menghindari hambatan mempertimbangkan setiap kendala yang pada gilirannya mungkin menggunakan skema portioning spasial untuk menyisihkan jarak agar keluar dari hambatan dan menentukan apakah karakter bola bersinggungan dengan silinder (Reynolds, 2010).



Gambar 4. Obstacle Avoidance (Reynold, 2010)

1.3 Collision Detection

Collision detection atau pendeteksian tumbukan adalah proses pengecekan apakah beberapa buah objek spasial saling bertumbuk atau tidak. Jika ternyata ada paling sedikit dua buah objek yang bertumbuk, maka kedua objek tersebut dikatakan saling bertumbukkan. Pada ruang spasial dua dimensi objek yang bertumbuk berarti objek spasialnya beririsan. Teknik pendeteksian tumbukan bisa dikelompokkan menjadi dua macam yaitu *priori detection* dan *post detection*. *Priori detection* adalah pengecekan tumbukan sebelum tumbukan tersebut terjadi, sedangkan *post detection*

adalah pengecekan tumbukan setelah tumbukan tersebut terjadi. (Maulana,2010)

Beberapa algoritma *collision detection* (Compsci, 2011):

1. *Basics: Simple collision detection*
 - *Rectangle – rectangle*
 - *Circle – circle*
 - *Circle - rectangle*
2. *Intermediate*
 - *Line – line*
 - *Circle – line*
 - *Bounding boxes*
3. *Advanced*
 - *Arbitrary polygonal shapes*
 - *Collision detection* berdasarkan waktu untuk mencegah *over lap* dan meningkatkan *precision*

Pada penelitian ini, akan menerapkan algoritma *Collision Detection (Circle – rectangle dan circle - line)*.

1.4 Algoritma Pathfinding

Tujuan dari algoritma *pathfinding* adalah untuk menemukan jalur terbaik dari vertex awal ke vertex akhir. Secara umum algoritma *pathfinding* digolongkan menjadi dua jenis (Stuart Russel dan Peter Norvig, 1995), yaitu :

1. *Algoritma Uniformed Search. Algoritma uniformed search* adalah algoritma yang tidak memiliki keterangan tentang jarak atau biaya dari path dan tidak memiliki pertimbangan akan path mana yang lebih baik. Yang termasuk dalam algoritma ini adalah algoritma *Breadth-First Search*.
2. *Algoritma Informed Search. Algoritma informed search* adalah algoritma yang memiliki keterangan tentang jarak atau biaya dari *path* dan memiliki pertimbangan berdasarkan pengetahuan akan *path* mana yang lebih baik. Yang termasuk algoritma ini adalah algoritma Dijkstra dan algoritma A*.

1.4.1 Algoritma A*

Dalam ilmu komputer, metode A* (*A Star*) adalah *graph search algorithm* yang mencari *path* (jalur) dari titik awal yang diberikan menuju titik tujuan.

A* pertama kali dijabarkan oleh Peter Hart, Nils Nilsson dan Bertram Raphael pada tahun 1968. (Wikipedia, 2014) Metode A* adalah metode yang merupakan hasil pengembangan dari metode dasar *Best First Search*. Metode ini mengevaluasi setiap titik dengan mengombinasikan dengan $g(n)$, nilai untuk mencapai titik n dari titik awal, dan $h(n)$, nilai perkiraan untuk mencapai tujuan dari titik n tersebut.

$$f(n) = g(n) + h(n)$$

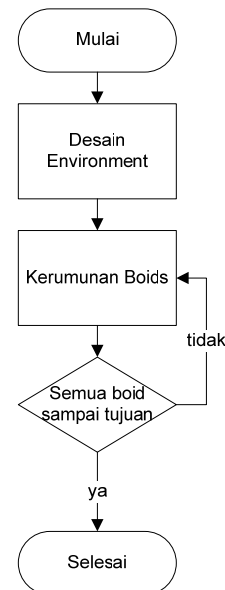
Ketika $g(n)$ memberikan hasil evaluasi nilai untuk mencapai titik n , dan $h(n)$ memberikan nilai estimasi untuk mencapai tujuan dari titik n , maka didapatkan $f(n)$ = nilai estimasi yang terkecil yang melewati titik n .

2. Metode

2.1 Pembuatan Model Simulasi

Pembuatan model simulasi dari algoritma *Boids* menggunakan suatu bahasa pemrograman html dan javascript, dalam hal ini hanya menggunakan pemodelan dua dimensi.

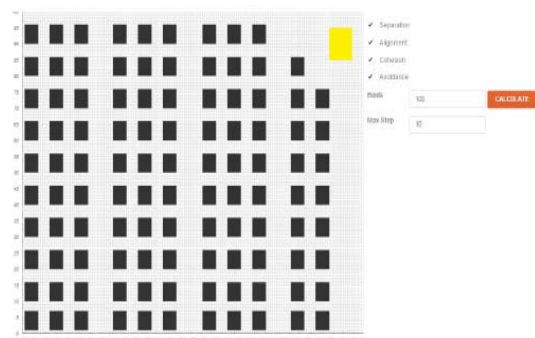
Adapun *flowchart* dari sistem simulasi ini dapat dilihat pada gambar 5. dibawah ini.



Gambar 5. Flowchart sistem simulasi

2.2 Desain Environment

Desain *Environment* terbatas yang dibatasi oleh dinding pembatas pada layar tampilan, sehingga sekelompok orang yang bergerak ke target utama akan menentukan jalur masing-masing dengan terlebih dahulu menentukan jarak terpendek dengan target utama. Setiap orang akan bergerak kearah kerumunan untuk bisa menuju target utama. *Environment* dirancang memiliki satu target utama.



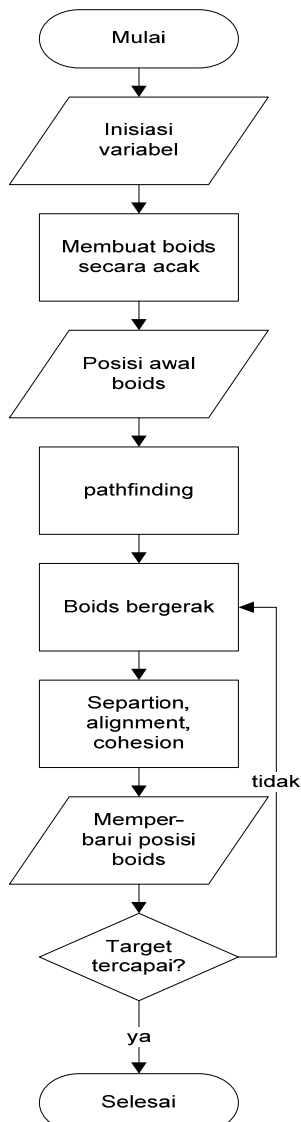
Gambar 6. Desain Environment

Desain *Environment* dibuat dengan menerapkan kerumunan orang-orang yang bergerak bebas dan acak memiliki target tertentu. Kerumunan dapat dilakukan

setelah arah masing-masing kelompok berdasarkan karakteristik diidentifikasi. Volume kerumunan tampak seperti kumpulan partikel fluida yang bergerak sesuai dengan kekuatan eksternal yang mendorong mereka

2.3 Pembuatan Algoritma *Boids* dan *Pathfinding*.

Adapun pembuatan Algoritma *boids* dengan menggunakan *flocking*, *obstacle avoidance*, *collision detection* dengan *pathfinding* dibuat dengan menerapkan keramaian orang-orang yang bergerak secara bebas dan acak yang mempunyai target tertentu. Volume kerumunan tampak seperti koleksi fluida partikel yang bergerak menuju tujuan target utama yang sama. Target adalah area yang harus dituju oleh semua orang. Pada simulasi ini, target pergerakan pengunjung berupa area titik aman tertentu, dapat dilihat pada *flowchart*.



Gambar 6. *Flowchart* algoritma *boids* dan *pathfinding*

Algoritma *boids* meliputi menjaga jarak agar menghindari terjadinya tabrakan antar agen (tetangga) dalam suatu kelompok tertentu yang mengarahkan *boids* bergerak menghindari kondisi yang padat agar tetap terpisah pada jarak pisah tertentu, *boids* berusaha menyesuaikan kecepatan (arah, kecepatan bergerak) dengan kecepatan agen tetangganya untuk bergerak ke arah tujuan kelompok yang sama dan mengarahkan agen ke titik pusat kelompok untuk tetap dekat dengan kelompoknya dengan mengarahkan *boids* bergerak menuju arah yang merupakan tujuan utama.

3. Hasil dan Pembahasan

Dalam tulisan ini, simulasi dibuat menggunakan html dan javascript. Dirancang untuk menempatkan posisi setiap penduduk dengan arah pergerakan bebas acak.. Posisi setiap orang tidak ada dan tidak boleh menempati area yang berwarna hitam. Sebuah target ditempatkan di salah satu sudut environment. Gerakan untuk menerapkan kelompok *boids* dengan tiga aturan: *alignment*, *separation* dan *cohesion*. Seluruh (kerumunan) bergerak menuju target dengan kecepatan tertentu. Hasil simulasi dapat dilihat pada Gambar 7.



Gambar 7. Simulasi Pergerakan Evakuasi

3.1 Analisa Hasil Pergerakan Populasi Menuju Target

Analisa pencapaian hasil simulasi tentang bagaimana pergerakan kerumunan orang menuju satu target utama diperoleh dengan melakukan uji coba pengaruh waktu yang diperlukan untuk mencapai target utama terhadap jumlah populasi kerumunan.

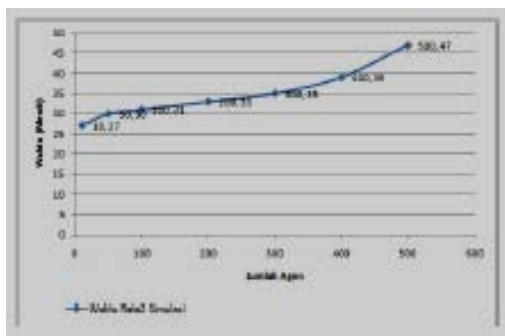
Pada simulasi menggunakan kecepatan (*velocity*) agen = 12 pixel/detik. Simulasi dilakukan pada lebar layar : 150 pixel dan tinggi = 100 pixel. Simulasi ini mewakili ruang sebenarnya yang berukuran 900 x 600 m, dengan kecepatan 1.2 m/detik. Data hasil pengukuran pengaruh waktu yang diperlukan untuk mencapai target utama terhadap jumlah populasi kerumunan dapat dilihat pada Tabel 1.

Terlihat dari hasil tabel tersebut, semua orang dalam berbagai jumlah populasi yang berbeda-beda di dalam area evakuasi, berhasil menuju target utama tanpa ada satupun yang tertinggal. Namun, semakin banyak atau semakin meningkat jumlah populasi kerumunan orang atau penduduk dalam area maka akan semakin banyak waktu yang diperlukan oleh kerumunan tersebut untuk mencapai target utama.

Tabel 1: Pengukuran Pengaruh Waktu terhadap Jumlah Populasi

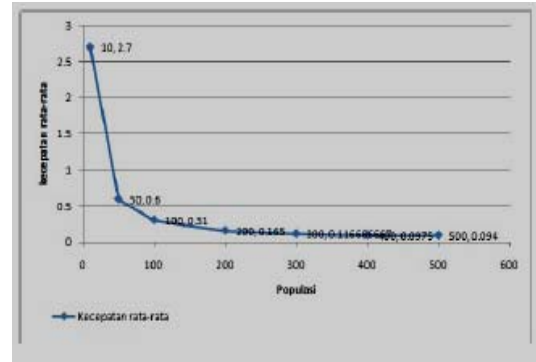
Simulasi ke-	Ujicoba ke-	Jml Populasi	Jml sampai	Waktu rata2 (menit)	kecepatan rata-rata
1	1	10	10		2.7
2	2	10	10		
3	3	10	10		
4	4	10	10		
5	1	50	50	30	0.6
6	2	50	50		
7	3	50	50		
8	4	50	50		
9	1	100	100	31	0.31
10	2	100	100		
11	3	100	100		
12	4	100	100		
13	1	200	200	33	0.165
14	2	200	200		
15	3	200	200		
16	4	200	200		
17	1	300	300	35	0.116667
18	2	300	300		
19	3	300	300		
20	4	300	300		
21	1	400	400	39	0.0975
22	2	400	400		
23	3	400	400		
24	4	400	400		
25	1	500	500	42	0.094
26	2	500	500		
27	3	500	500		
28	4	500	500		

Dari data hasil pengukuran pengaruh waktu yang diperlukan untuk mencapai target utama terhadap jumlah populasi kerumunan orang pada Tabel 1, dapat diperlihatkan dengan grafik pada Gambar 8. Dari grafik tersebut terlihat bahwa, ke naikkan waktu akan terus bertambah seiring bertambahnya jumlah populasi kerumunan penduduk untuk bergerak mencapai target utama.



Gambar 8. Waktu Rata-Rata Pergerakan Orang Terhadap Jumlah Populasi Kerumunan

Dari data pada Tabel 1, dapat diperlihatkan pada grafik Gambar 9, pengaruh kecepatan rata-rata masing- masing orang mencapai target utama dipengaruhi dengan bertambahnya jumlah populasi di dalam area. Semakin meningkatnya jumlah populasi, akan menurunkan tingkat kecepatan rata-rata per orang untuk bergerak mencapai target utama.



Gambar 9. Grafik Kecepatan Rata-Rata Pergerakan Orang Terhadap Jumlah Populasi Kerumunan

3.2 Analisa Pencapaian Hasil Persentase Jumlah Penduduk yang berhasil menuju target utama

Berdasarkan Buku Masterplan Pengurangan Resiko Bencana Tsunami (BNPB, 2012), mengenai waktu evakuasi minimal yang diperlukan oleh orang-orang yang berada dalam wilayah jangkauan tsunami adalah dalam jangka waktu maksimal 59 menit setelah terdengarnya sirene peringatan dini. Mengacu pada hal tersebut, pada simulasi ini ditetapkan waktu evakuasi maksimal untuk para penduduk mencapai target utama jika terjadi keadaan darurat. Data hasil pengukuran diperlihatkan pada Tabel 2.

Tabel 1: Pengukuran Persentasi Jumlah Penduduk Terperangkap dan Berhasil Menuju Target

Jumlah Populasi	Tidak Berhasil ke tujuan	Berhasil ke tujuan	Persentasi Berhasil
10	0	10	100
50	0	50	100
100	0	100	100
200	0	200	100
300	0	300	100
400	0	400	100
500	0	500	100

4. Kesimpulan Hasil Simulasi

Penggunaan algoritma *boids* dan *pathfinding* berhasil diterapkan sebagai algoritma untuk simulasi kerumunan menuju target tertentu. Peubah simulasi adalah jumlah penduduk dan kecepatan rata-rata menuju target. Pertambahan jumlah kerumunan membutuhkan selang waktu yang lebih menuju target yang diinginkan.

Algoritma *boids* dan *pathfinding* berhasil memenuhi waktu maksimal yang diperlukan untuk melakukan evakuasi yaitu kurang dari 59 menit.

Ucapan Terima Kasih

Terima kasih kepada semua pihak yang mendukung saya untuk menyelesaikan paper ini walaupun masih banyak kekurangan. Terutama untuk Ida Sang Hyang Widhi Wasa, Kominfo, Jurusan Teknik Elektro, Mauridhi Hery Purnomo, Supeno Mardi Susiki Nugroho, Kawan-kawan CIO ITS Angkatan 2013.

Daftar Pustaka

- Abrahams, John, (1994) , Fire escape in difficult circumstances ,chapter 6, In: Stollard
- Aspelin, K, (2005), Establishing Pedestrian Walking Speeds, Karen Aspelin, P.E.,P.T.O.E., ITE District 6 Technical Chair Parsons Brinckerhoff Albuquerque, New Mexico. Portland State University.
- BNPB (2012) Masterplan Pengurangan Resiko Bencana Tsunami
- CompSci, (2006), Collision Detection Tutorial, at <http://compSci.ca/v3/viewtopic.php?t=13661> [Maret 2014]
- Cui, X, (2006), A Flocking Based Algorithm for Document Clustering Analysis, Journal of System Architecture
- Dewi, Meilany, (2012), Simulasi Pergerakan Pengunjung Mall Menggunakan Flocking dan Obstacle Avoidance, Institut Teknologi Sepuluh Nopember, Tesis
- Maulana, S, N, (2010), Penggunaan Struktur Data Quad-Tree dalam Algoritma Collision Detection pada Vertical Shooter Game,Makalah IF3051 Strategi Algoritma Sem.I Tahun 2010/2011, Institut Teknologi Bandung, Bandung
- Reynolds, C.W, (2010), Steering Behaviors For Autonomous Characters, <http://www.red3d.com/cwr/steer/gdc99/> [Maret 2014]
- Stuart Russell, Peter Norvig, (1995), Artificial intelligence : a modern approach Prentice-Hall, Inc. A Simon and Schuster Company, Englewood Cliffs, New Jersey 07632
- Undang-undang Nomor 24 Tahun (2007), Pengulangan Bencana
- Wikipedia, (2014), A* search algorithm, at http://en.wikipedia.org/wiki/A*_search_algorithm [Juni 2014]