SISTEM PENENTU GERAKAN MOBILE ROBOT YANG BELAJAR SENDIRI CARA UNTUK BERGERAK MAJU DAN MENGINDARI TRABRAKAN MENGGUNAKAN *NEURAL NETWORK* (NN)

Eru Puspita (eru@eepis-its.edu)
Politeknik Elektronika Negeri Surabaya
Institut Teknologi Sepuluh Nopember
Kampus ITS Keputih Sukolilo Surabaya 60111, Indonesia

ABSTRACT

Commonly used for driving mobile robot is by using certain pre defined algorithm. This research will try to develop driving technique using continuous self learning NN while mobile robot on go. This research only observe about how mobile robot by it self can move forward an how to avoid a collision. Program NN will try to learn how to move forward, backward, turn left, turn right or other possibility depend on or by using collision experience. The test results obtained 82% successful runs forward and approximately 90% managed to avoid a collision.

Keywords: kendali gerakan,mobile robot, NN, on-line

Sistem kendali gerakan mobile robot yang digunakan dewasa ini banyak yang menggunakan pengaturan gerakan secara langsung berdasarkan informasi dari sensor dan dari sasaran gerakan yang diinginkan. Seluruh proses pergerakan dari motor-motor penggeraknya diatur sepenuhnya oleh program atau *hardware*. Dengan cara ini maka tingkat kepastian gerakan akan sesuai dengan yang diinginkan.

Penelitian yang terkait dengan masalah pengendalian gerakan mobile robot memiliki berbagai kelompok tujuan utama, seperti Hachour (2008) dan Dan (2009) yang menyajikan tentang teknik penentuan lintasan, baik untuk menentukan lintasan bergeraknya atau untuk menghindari halangan. Kumar et al (2010) membahas khusus cara untuk menghindari halangan atau Borenstein dan Koren (1987) yang membahas teknik untuk menjaga agar lintasan atau posisi dijaga tepat dengan menghindari slip dan memperkecil kesalahan posisi atau Lin dan Yang (2008) yang membahas teknik pengendalian lintasan dan kecepatan. Sebagian lagi melakukan penelitian selain teknik menghindari halangan juga mengangkat masalah *local minima* yang disebabkan oleh halangan dalam bentuk lingkungan.

Beberapa penelitian seperti yang dilakukan oleh Doitsidis et al (2002), Kim dan Mohan Trivedi (1998), Singh, dan Parhi (2009), Kumar (2009), Tahboub dan Munaf (2009), Velagic et al (2008), Pennacchio, S dkk (2005) serta penelitian Gavrilov dan Lee (2007) menerapkan kecerdasan buatan untuk menghindari tabrakan baik yang murni menggunakan aturan *Fuzzy Logic* atau kombinasi *Neuro-Fuzzy* menggunakan proses pembelajaran awal atau pembelajaran sambil jalan.

Pennacchio et al (2005) pada makalahnya menyodorkan suatu metode yang dinamakan FULURO (*Fuzzy Logic Robot*), yaitu kombinasi antara *Fuzzy Logic* dan NN untuk menjalankan robot. *Fuzzy Logic* digunakan untuk membaca sensor-sensor halangan dan sensor kecepatan dua roda untuk menentukan perilaku (bisa dianggap menentukan arah yang harus diambil dan kecepatan) dari

mobile robot. Keluaran dalam bentuk perilaku ini digunakan untuk pelatihan NN secara on-line menggunakan pelatihan *BP* untuk meningkatkan solusi perilaku.

Gavrilov dan Lee (2007) pada makalahnya menyajikan cara untuk menjalankan robot secara *semi supervised*, dimana robot sedikit diajari saat robot tidak dapat mengenali informasi yang didapat agar robot dapat mengambil tindakan yang benar, selebinya tergantung dari robot untuk memutuskan tindakan yang harus dilakukan. Program sistem ini menggunakan kombinasi Neural Network (NN) jenis MLP-ART2.

Pada penelitian ini akan dicoba untuk membangun sistem kendali gerakan berdasarkan proses belajar dari sistem itu sendiri menggunakan NN sebagai pengendali gerakannya. Hal ini didasari oleh NN yang memiliki kemampuan untuk belajar, sehingga pada penelitian ini ingin dibuktikan apakah NN benar-benar dapat belajar untuk mengendalikan gerakan dari mobile robot berdasarkan informasi dari sensor-sensor yang ada.

Ada beberapa cara yang dapat digunakan untuk mengendalikan gerakan mobile robot menggunakan NN, antara lain:

a. Pengendalian secara *off-line*, dimana NN diberikan pelatihan terlebih dulu bagaimana cara untuk bergerak maju dan menghindari halangan. Cara ini memungkinkan NN untuk tahu lebih dulu cara bergerak sebelum diujikan pada lapangan.
b. Pengendalian berdasar pemberian referensi gerak yang dilakukan secara *on-line*. Cara ini mirip dengan pengendalian konvensional, namun sifatnya NN hanya diberikan perintah untuk bergerak ke mana tanpa memberitahukan cara bergeraknya.
c. Pengendalian berdasarkan menunjukkan bagian yang salah (*error*) dari sistem mobile secara *on-line*. Misalkan menunjukkan roda mana yang salah bergerak. Cara ini mirip dengan cara b, namun NN tidak secara langsung diberi perintah harus bergerak ke mana.
d. Pengendalian berdasarkan hanya menunjukkan kalau ada yang salah (*error*), tanpa menunjukkan apa atau dimana yang salah. Cara ini lebih sulit, karena mengharuskan NN untuk mencari kemungkinan-kemungkinan bagian mana yang salah.

Pada penelitian ini akan dicoba cara pada point c. Tujuan yang ingin dicapai dalam penelitian ini adalah: Pertama, agar NN dapat digunakan untuk mengendalikan gerakan dari mobile robot berdasarkan informasi dari sensor. Kedua, agar NN dapat belajar dengan sendirinya secara on-line berdasarkan informasi kesalahan yang diterima. Dan tujuan ketiga adalah agar mobile robot dapat melaju ke depan dengan berusaha menghindari halangan yang ada.

Dalam penelitian ini dicari cara untuk memformulasikan informasi dari sensor agar dapat digunakan sebagai sarana belajar sendiri bagi NN, sehingga NN dapat mengendalikan gerakan mobile robot agar dapat maju tanpa menabrak penghalang, penelitian hanya dibatasi agar mobile robot dapat melaju ke depan dengan berusaha menghindari halangan yang ada, dan pada penelitian awal ini hanya diujikan pemodelan sistem kendali secara simulasi.

METODE PENELITIAN

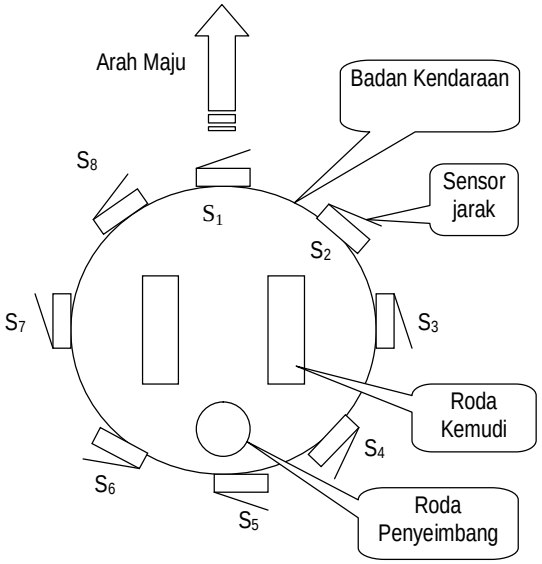
Sebelumnya telah dilakukan pengujian penggunaan kemampuan NN untuk menerima masukan dari sensor sehingga NN dapat memutuskan untuk melakukan sesuatu seperti yang diinginkan. Pengujian pertama yang dilakukan adalah menggunakan NN sebagai kontrol posisi suatu mobile robot pada suatu dinding. Di sini NN harus menjaga jarak mobile robot pada jarak tertentu dengan dinding. Di awal, sensor yang digunakan adalah *limit switch*. Pengujian ini gagal, karena NN

tidak dapat menerima masukan dalam bentuk biner (on/off atau menabrak/tidak menabrak). Hal ini dikarenakan NN hanya dapat belajar dari membandingkan satu kesalahan dengan kesalahan lainnya, tetapi dengan nilai yang tidak boleh terlalu ekstrim. Pengujian dilanjutkan dengan menggunakan sensor jarak, sehingga NN dapat mengukur jarak mobile robot dengan dinding. Pada pengujian ini, NN dapat melakukan tugasnya dengan baik.

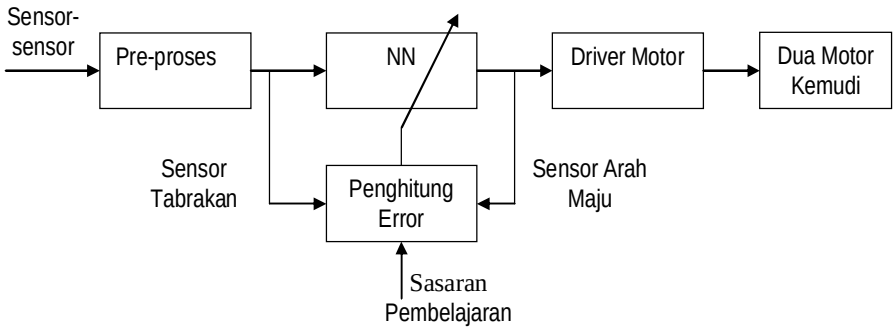
Pengujian kedua dilakukan dengan mencoba menggunakan jumlah masukan untuk informasi error yang tidak sama dengan jumlah keluaran dari NN. Umumnya NN menggunakan informasi error dengan jumlah yang sama dengan keluaran dan referensi untuk NN. Dari uji coba ini dihasilkan bahwa NN dapat digunakan, asalkan dapat dibuat suatu formula yang dapat digunakan untuk mengubah jumlah informasi error yang ada menjadi sejumlah tertentu yang sama dengan output NN.

Dari dua percobaan sederhana tersebut dicoba untuk merancang sistem pengendalian gerakan mobile menggunakan NN yang dapat belajar sendiri berdasarkan informasi dari beberapa sensor untuk menggerakkan dua motor penggerak agar mobile robot dapat bergerak maju dan dapat menghindari tabrakan. Secara umum konfigurasi yang diinginkan adalah seperti pada Gambar 1 dan Gambar 2. Metode yang digunakan dalam penelitian ini secara umum menggunakan sistem yang sudah sering dijumpai, yaitu NN dengan pembelajaran *Back-propagation*. Dua hal atau sub-sistem yang ditambahkan yaitu penghitung error dan pengereman. Karena itu, dalam pembahasan berikutnya tidak akan banyak ditampilkan formula-formula, kecuali untuk penghitung error dan pengereman, serta penjelasan konfigurasi sistem yang dibuat. Selebihnya (misalkan tentang NN) sudah banyak dibahas dalam buku text seperti pada Freeman et al (1992) atau referensi lain.

Untuk sensor, pre-proses, *driver*, motor, mekanik lainnya tidak akan dibahas dalam artikel ini, karena tergantung dari jenis yang digunakan atau tergantung kebutuhan. Yang akan dibahas hanya mekanisme agar suatu mobile robot dapat memiliki kemampuan bergerak maju dan menghindari tabrakan dengan sendirinya, melalui proses belajar sendiri tanpa menggunakan panduan dari luar atau proses pembelajaran sebelumnya secara *off-line*.



Gambar 1. Konfigurasi mekanik robot



Gambar 2. Konfigurasi sistem

Gambar 1. menjelaskan bentuk mekanik robot dan Gambar 2 menunjukkan konfigurasi sistem kontrol menggunakan NN dan proses pembelajaran secara *on-line*.
Cara kerja robot dapat dijelaskan sebagai berikut: Pertama kali, program tidak memiliki kemampuan mengemudi sama sekali, atau bisa juga mengambil nilai kemampuan yang telah disimpan sebelumnya. Jika program dalam keadaan tidak memiliki kemampuan sama sekali (baru pertama dioperasikan), maka NN akan mengeluarkan tegangan kontrol ke motor secara acak dan tak terkendali. Untuk mencegah hal yang tidak diinginkan, dalam keadaan belum memiliki kemampuan mengemudi, dilengkapi dengan pembatas kecepatan (sistem pengereman) motor. Berdasarkan sensor tabrakan, sensor arah maju dan sasaran pembelajaran yang telah ditentukan (dalam hal ini program diperintahkan untuk menghindari dari tabrakan dan bergerak maju), maka program menghitung nilai error yang terjadi. Setiap terjadi tabrakan, atau motor tidak bergerak maju, maka dianggap ada suatu kesalahan. Besarnya nilai error digunakan untuk mempengaruhi NN agar melakukan proses pembelajaran (mengubah bobot-bobot dalam NN). Selama proses pembelajaran, NN akan mengeluarkan tegangan-tegangan kontrol ke motor yang berubah-ubah dan sulit ditentukan keadaannya. Dari sini nantinya ingin dilihat, apakah NN dapat belajar untuk maju dan menghindari tabrakan dengan caranya sendiri.

Jika proses pembelajaran berhasil, NN akan mengeluarkan tegangan kontrol ke motor yang menyebabkan motor bergerak untuk menghindari dari tabrakan sambil berusaha maju ke depan.

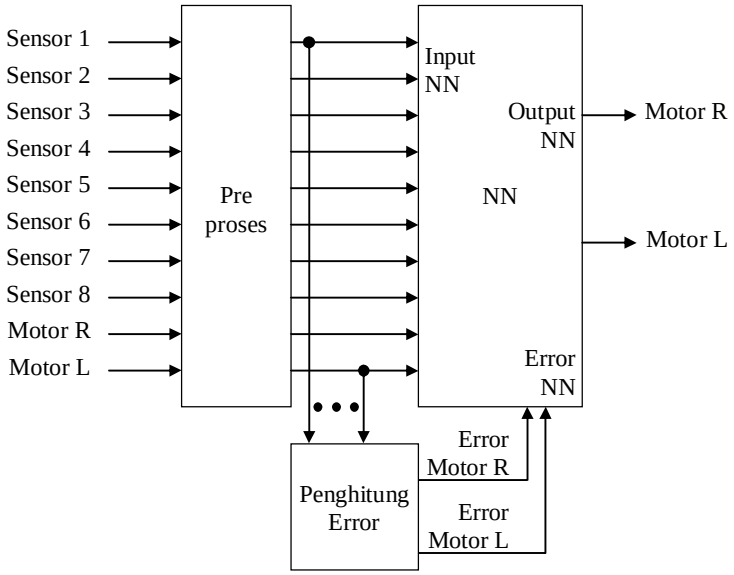
Arsitektur NN untuk kontroler

Gambar 3 memperlihatkan arsitektur NN dalam percobaan. Dalam percobaan ini digunakan NN dengan konfigurasi 10 neuron pada input layer untuk membaca 8 sensor tabrakan dan dua sensor kecepatan, beberapa (10 sampai 20) neuron pada *hidden layer* dan 2 neuron pada output layer untuk menjalankan motor kiri (motor L) dan motor kanan (motor R).

Sensor tabrakan adalah sensor jarak yang digunakan untuk mengukur jarak benda yang ada di depan sensor. Sensor ini direncanakan memiliki jangkauan 0 sampai dengan 25 cm. Sedangkan sensor kecepatan digunakan untuk mengetahui besarnya kecepatan motor saat itu, termasuk maju atau mundur. Pre-proses hanya digunakan untuk mengkondisikan keluaran dari sensor tabrakan dan sensor kecepatan agar sesuai dengan masukan dari NN yaitu bernilai 0 sampai dengan 1.

Kunci utama dari sistem ini adalah model atau formula yang ada pada penghitung error. Dalam model NN, jumlah error yang dimasukkan atau akan digunakan dalam proses pembelajaran adalah sama dengan jumlah neuron keluarannya. Di mana error berarti error keluaran dari NN terhadap nilai referensi. Namun dalam sistem ini, sama sekali tidak digunakan atau tidak ada data

atau nilai referensi yang diberikan. Tentu saja hal ini akan menyulitkan proses pembelajaran NN. Satu-satunya informasi yang dapat digunakan untuk proses pembelajaran adalah input yang sejumlah 10 titik tersebut, yang sebenarnya input ini tidak sesuai untuk error baik dari sisi hubungan atau jumlah inputnya. Penghitung error dianggap sebagai penentu utama karena dia harus dapat mengubah atau menghitung dari 10 input menjadi 2 data error yang nantinya dapat digunakan untuk proses pembelajaran NN.



Gambar 3. Arsitektur NN untuk sistem kontrol

Mengingat tidak ada relasi yang langsung secara matematik antara input dan error, maka dibuat pendekatan fungsional (operasional), yaitu kalau suatu sensor memiliki nilai tertentu atau mendeteksi kondisi tertentu, maka yang harus disalahkan adalah motor yang sebelah mana. Dari pendekatan yang sederhana ini dapat dibuat formula yang sangat sederhana dari penghitung error dan bahkan bentuk formulanyapun dapat dibuat beragam, asalkan memenuhi pendekatan yang telah digunakan.

$$e_L = (-S_1 - S_2 + S_7 + S_6 + S_5 + sM_L)$$

$$e_R = (-S_1 - S_8 + S_3 + S_4 + S_5 + sM_R) \quad (1)$$

atau

$$e_L = (-S1 - S2 + S7 + S6 + S5 + sM_L) + (Rnd * 0.5 - 0.25)$$

$$e_R = (-S1 - S8 + S3 + S4 + S5 + sM_R) + (Rnd * 0.5 - 0.25) \quad (2)$$

dimana:

e_L dan e_R adalah error keluaran penghitung error untuk proses belajar NN

S_1 s/d S_8 adalah keluaran sensor jarak setelah melalui pre-proses

sM_L dan sM_R adalah keluaran sensor kecepatan setelah melalui pre-proses

Formula (1) adalah bentuk yang paling sederhana untuk menentukan kesalahan gerak dari motor. Sedangkan formula (2) ditambahkan fungsi random untuk mendapatkan kemungkinan gerakan yang beragam, atau dapat digunakan bentuk-bentuk formula lainnya.

On-line Update

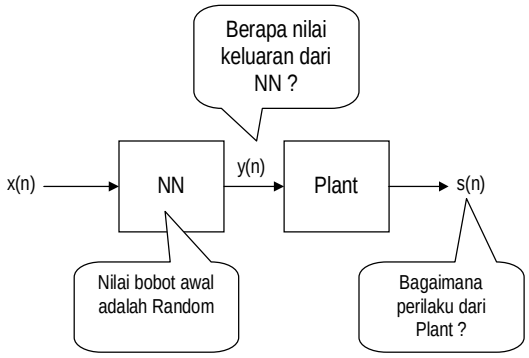
Umumnya NN dilatih dengan cara diberikan contoh-contoh masukan dan keluaran yang seharusnya dikeluarkan NN (*mode supervised*). Jika antara keluaran NN dan keluaran yang diinginkan tidak sesuai, maka akan dihitung error yang nantinya error ini digunakan sebagai proses update. Jika dilakukan secara terus menerus, maka NN akan dapat menyesuaikan keluarannya sesuai dengan yang diinginkan. Proses ini dikenal dengan nama pelatihan (*training*) dan biasanya dilakukan terlebih dahulu sebelum NN digunakan.

Tujuan dari cara tersebut adalah mempersiapkan agar NN memiliki karakteristik sesuai dengan yang diinginkan sebelum NN digunakan. Cara ini akan menjamin NN tidak akan memiliki perilaku atau karakteristik yang tidak terprediksi yang dapat membahayakan sistem. Namun jika dapat dipastikan sistem akan mentolerir atau perilaku tidak terduga dari NN tidak akan membahayakan sistem, maka proses pembelajaran atau pelatihan NN dapat dilakukan sekaligus saat NN digunakan atau dijalankan. Cara ini disebut sebagai *On-line Training* atau *Learning* atau *Update*.

Jika memungkinkan, maka cara ini akan lebih praktis, karena proses dapat dilakukan sekali jalan, yaitu jalan dan belajar. Selain itu dimungkinkan melakukan adaptasi (proses untuk sedikit belajar mengikuti perubahan karakteristik sistem) secara langsung, yang tidak dapat dilakukan pada cara off-line (pelatihan yang terpisah).

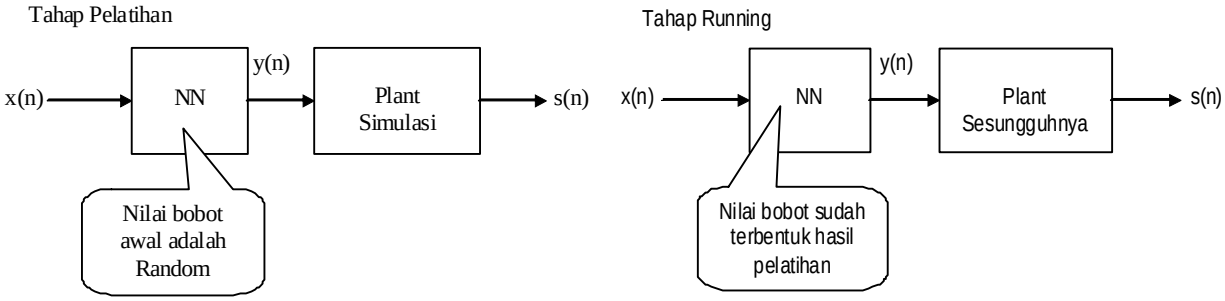
Sistem Pengaman Keluaran NN

Dalam keadaan awal NN bekerja, umumnya bobot-bobot NN ditentukan secara acak. Jika demikian, apabila masukan NN diberikan nilai tertentu, maka berapakah nilai keluaran dari NN? Jawaban pastinya adalah tidak dapat diketahui atau acak. Dalam keadaan seperti ini, jika NN digunakan untuk menggerakkan suatu *plant* secara langsung, maka dapat berakibat timbulnya hal-hal yang tidak dapat ditentukan yang dapat membahayakan plant itu sendiri.



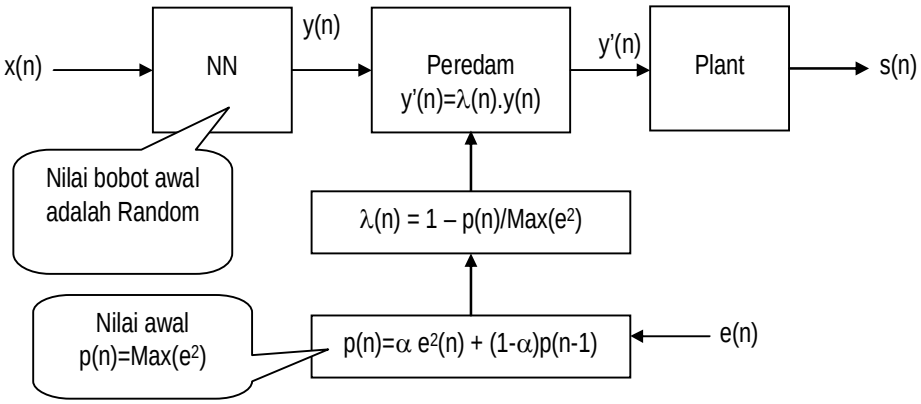
Gambar 4. Ilustrasi keadaan awal NN

Cara yang umum dilakukan adalah menggunakan pembelajaran awal (*off-line training*), dimana NN diberikan plant dalam bentuk miniatur atau simulasi, sehingga keadaan awal dari NN yang dapat membahayakan plant dapat dihindari. Namun cara ini tidak praktis, karena harus menggunakan dua tahap operasional, *training* dan *running*.



Gambar 5. Teknik mengatur keluaran dari NN menggunakan pelatihan awal

Mengingat dalam penelitian ini diinginkan untuk proses belajar secara on-line, maka di sini ditambahkan sub-sistem pengereman, dimana sub-sistem ini akan meredam keluaran dari NN jika NN masih memiliki rata-rata error yang besar. Nilai peredaman bergantung dengan besar error dari NN. Dalam keadaan awal, nilai error dapat dianggap besar, sehingga peredaman ini dapat mencegah perilaku awal dari NN yang acak, tidak langsung masuk ke plant.



Gambar 6. Teknik mengatur keluaran dari NN menggunakan pengereman

Pada penelitian ini digunakan formula pengereman seperti pada formula (3).

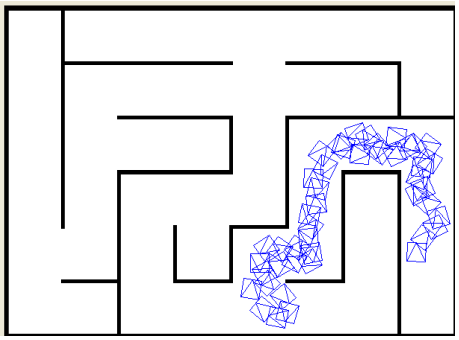
$$e = Abs(e_R) + Abs(e_L)$$
$$M_R = (nO_R * 2 - 1) * (2 - Abs(e)) * 2$$
$$M_L = (nO_L * 2 - 1) * (2 - Abs(e)) * 2 \quad (3)$$

Dimana:

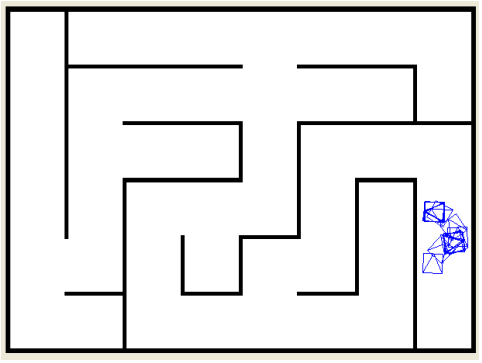
e_R dan e_L adalah error hasil keluaran penghitung error
M_R dan M_L adalah keluaran untuk mengatur kecepatan motor kiri dan kanan
nO_R dan nO_L adalah keluaran dari NN untuk motor kiri dan kanan

HASIL DAN PEMBAHASAN

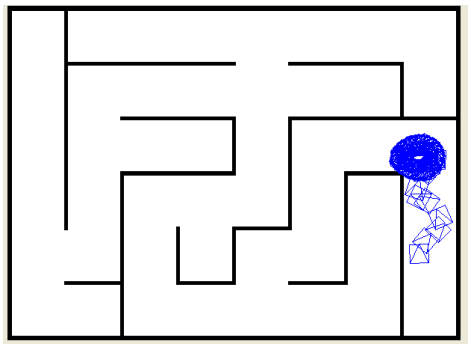
Hasil penelitian diberikan dalambentuk *screen shoot* pengujian NN secara simulasi menggunakan komputer, untuk memberikan gambaran kemampuan NN dalam menentukan pergerakan mobile robot, hanya berdasar informasi kesalahan yang diberikan. Pada pengujian diberikan arena dalam bentuk maze. Tugas utama dari NN adalah harus dapat menggerakkan mobile robot berjalan di sepanjang lorong-lorong yang ada tanpa menabrak dinding pembatas. Pengujian dibagi tiga tahap, tahap pertama adalah tahap awal pembelajaran NN, dimana nilai bobot dari NN masih acak. Pada pengujian ini akan didapatkan berbagai macam hasil yang tidak dapat diperkirakan. Dari gambar 7 sampai 10 dapat dilihat berbagai kemungkinan hasil pengujian saat awal pembelajaran. Pada Gambar 8 dan 9 mobile robot terjebak dalam jalur yang sempit, sehingga hanya berputar-putar pada tempat tersebut. Pada Gambar 7 dan 10 mobile robot berhasil mencapai jarak tempuh cukup jauh.



Gambar 7. Pembelajaran awal

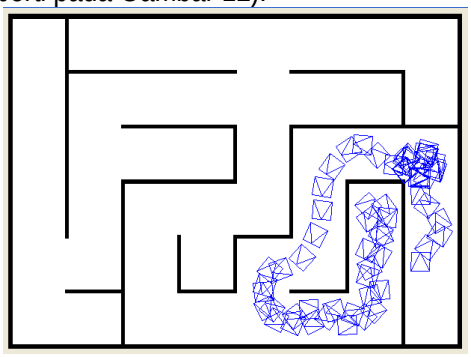


Gambar 8. Pembelajaran awal, tidak dapat bergerak

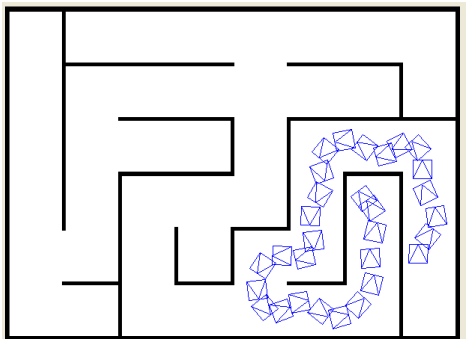


Gambar 9. Pembelajaran awal, berputar di satu lokasi

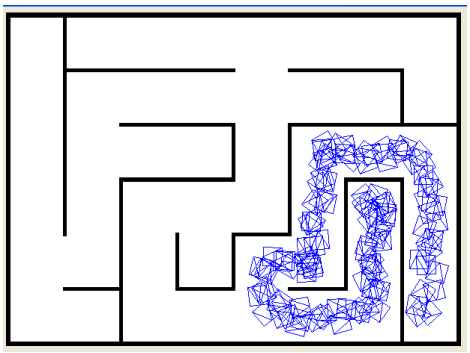
Pengujian kedua dilakukan saat iterasi mencapai lebih dari 10.000 kali seperti pada Gambar 11 posisi mobile robot dikembalikan ke posisi awal. Pada gambar tersebut terlihat, bahwa NN sudah tidak mengalami banyak kesulitan saat bergerak. Jika pengujian diteruskan sampai ribuan iterasi berikutnya, ternyata mobile robot hanya berputar-putar pada lokasi yang telah dikenal tersebut, tanpa bisa mencapai tempat lain (seperti pada Gambar 12).



Gambar 10. Pembelajaran awal, berhasil mencapai satu lokasi



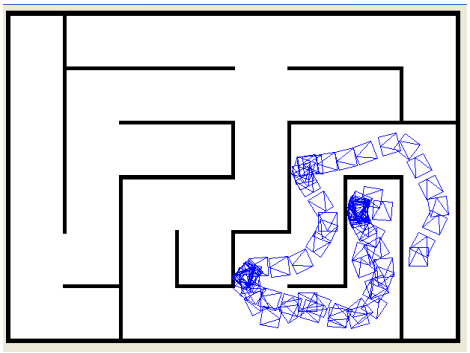
Gambar 11. Setelah iterasi lebih dari 10000, posisi dimulai dari awal



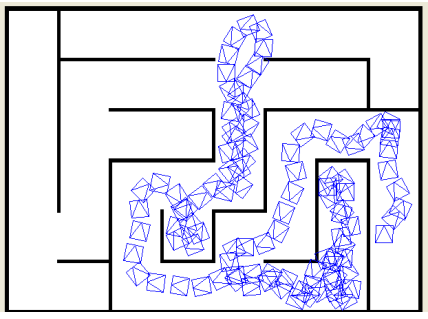
Gambar 12. Berputar di sekitar lokasi yang dikenali

Bahkan setelah mencapai puluhan ribu iterasi, mobile robot hanya dapat berputar pada lokasi yang sama. Pada pengujian ketiga, dicoba untuk mengubah beberapa parameter dari robot, antara lain target kecepatan menjadi 90% dari kecepatan tertinggi (Gambar 13). Memperbesar nilai error kecepatan 1,5 kali dari nilai semula (lebih besar 50%), seperti Gambar 13. Menambah kecepatan mobile robot sebesar 2 kali dan 4 kali dari nilai semula, seperti pada Gambar 15, 16, 17.

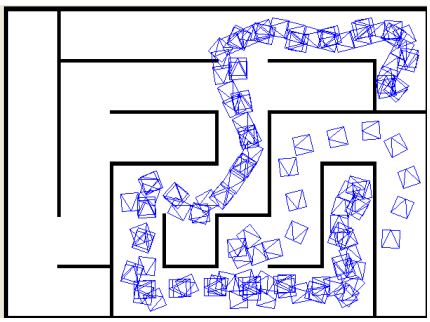
Penambahan kecepatan pada bagian akhir pengujian ternyata membawa dampak mobile robot mampu untuk menjangkau daerah-daerah lain yang lebih jauh. Namun yang perlu diingat, penambahan kecepatan ini hanya dapat dilakukan jika NN sebelumnya sudah belajar.



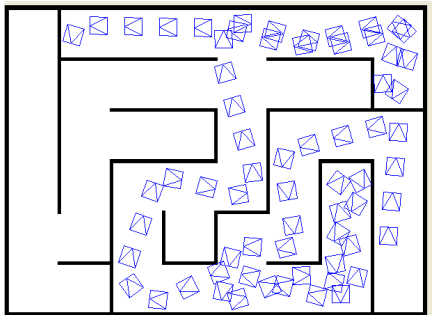
Gambar 13. Kecepatan maju ditingkatkan menjadi 90%



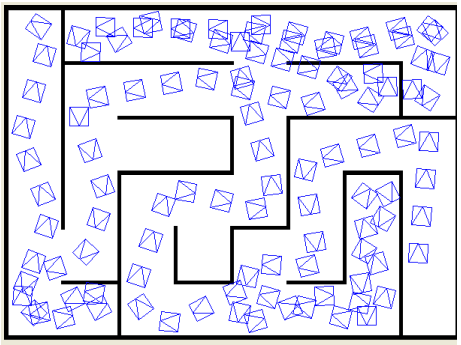
Gambar 14. Error kecepatan diperbesar 50%



Gambar 15. Menambah kecepatan motor 2 x



Gambar 16. Menambah kecepatan 4 x



Gambar 17. Hasil akhir pada kecepatan 4x

KESIMPULAN

Secara umum, jika target pengujian adalah mobile robot dapat bergerak terus ke depan, maka ditemukan 2 dari 11 pengujian, dimana mobile robot terjebak berputar-putar pada satu titik sempit. Pengamatan terhadap kemungkinan lebih dari 200 rawan tabrakan, didapatkan sekitar 20 kali robot menyentuh dinding pembatas, bahkan ditemukan mobile robot menerobos dinding pada pengujian kecepatan motor yang lebih tinggi. Asalkan yang diinginkan adalah kemampuan bergerak secara mandiri, metode yang digunakan dalam penelitian ini dapat dimanfaatkan.

REFERENSI

Borenstein, J. & Yoram, K. (1987). Motion control analysis Of A mobile robot. *Transactions of ASME, Journal of Dynamics, Measurement and Control*, Vol. 109(2), 73-79.

Dan, P.K. (2009). *Obstacle avoidance method for a mobile robot*. Thammasat Int. J. Sc. Tech., Vol. 14(4), October-December 2009.

Doitsidis, L., Valavanis, K. P., & Tsourveloudis, N.C. (2002). *Fuzzy logic based autonomous skid steering vehicle navigation*. Proceeding of the 2002 IEEE International Conference on Robotics & Automation Washington, DC May 2002.

Freeman, J.A. & Skapura, D.M. (1992), *Neural networks, algorithms, applications, and programming techniques*, New York: Addison-Wasley.

Gavrilov, A.V & Lee S. (2007). *An architecture of hybrid neural network based navigation system for mobile robot*. Proceedings of the Seventh International Conference on Intelligent Systems Design and Applications.Pages: 587-590.

Hachour, O. (2008). Path planning of Autonomous mobile robot. *International Journal Of Systems Applications, Engineering & Development.* 4(2).

Tahboub K.K., & Munaf. (2009). A neuro-fuzzy reasoning system for mobile robot navigation. *Jordan Journal of Mechanical and Industrial Engineering.* Volume 3(1), 77 – 88.

Kim C. Ng, & Mohan M.T. (1998). *A neuro-fuzzy controller for mobile robot navigation and multirobot convoying*. IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS—PART B: CYBERNETICS, Vol. 28(6), December 1998.

Kumar, P, Jindal. N, Jindal. A & Chhabra.S. (2010). A Robust Algorithm for Local Obstacle Avoidance. *International Journal of Computer Theory and Engineering*. June, 2010; Vol. 2(3).

Kumar, S.M, Parhi D.R., & Pothal J.K. (2009). *ANFIS approach for navigation of mobile robots*. International Conference on Advances in Recent Technologies in Communication and Computing.

Pennacchio, S, Abissi, F., Petralia S., & Stendardo, G. (2005). *New intelligent controller for mobile robot navigation in unknown environments*. Proceedings of the 6th WSEAS Int. Conf. on FUZZY SYSTEMS, Lisbon, Portugal, June 16-18, (pp79-82).

Singh, M.K., Dayal R. Parhi. (2009). *Intelligent neuro-controller for navigation of mobile robot*. Proceedings of the International Conference on Advances in Computing, Communication and Control, Mumbai, India, Pages: 123-128.

Lin S.W, & Yang P.C. (2008). Adaptive critic motion control design of autonomous wheeled mobile robot by dual heuristic programming. *Journal Automatica 44.*