Pemrograman Robotik Menggunakan Bahasa Java

(Studi kasus *Line Maze*)

Marojahan MT Sigiro[1], Eva M Siahaan[2], Jogi Silalahi[3], Olivia Irma Sari[4]

Politeknik Informatika Del

Jl. Sisingamangaraja, Sitoluama, Laguboti, TOBASA, 22381, Indonesia

Email: [1]marojahan@del.ac.id, {[2]if07068, [3]if07035, [4]if07034}@students.del.ac.id

Abstrak

Makalah ini disarikan dari Tugas akhir mahasiswa pidel[6]. Dalam makalah ini dilakukan pembahasan pemrograman robot dengan menggunakan java pada platform robot Lego Mindstorm NXT. Lego Mindstorms NXT adalah sebuah perangkat robotik yang memiliki brick yang dilengkapi dengan firmware yang dapat diprogram menggunakan aplikasi ROBOLAB. Untuk meningkatkan fleksibilitas robot lego, ada berbagai proyek pengembangan firmware yang menggantikan firmware lego dan salah satunya adalah leJOS. LeJOS memiliki fitur berupa Java Virtual Machine sehingga memungkinkan menjalankan program java pada robot Lego Mindstorm NXT. Penerapan pemrograman java pada robot lego pada penelitian ini dilakukan dengan membuat library gerakan primitif robot. Gerakan primitif yang dimaksud adalah gerakan dasar yang nantinya bisa digunakan untuk merangkai gerakan yang lebih kompleks. Dalam penelitian ini dibuat sebuah program menggunakan library gerakan primitif yang dihasilkan untuk menyelesaikan sebuah kasus line maze.

Kata kunci: Java, Lego mindstorm NXT, Robotik

1 Pendahuluan

Robot merupakan perangkat mekanik yang mampu menjalankan tugas fisik yang kompleks secara otomatis baik dibawah kendali dan pengawasan manusia ataupun yang dijalankan dengan serangkaian program yang telah didefinisikan terlebih dahulu atau kecerdasan buatan *(Artificial Iintelligence)* [4]. Faktor yang mendorong semakin berkembangnya pemrograman robotik antara lain disebabkan oleh perkembangan berbagai perangkat robot yang meyediakan fitur berupa API maupun fitur lainnya yang memungkinkan pengguna dapat mengkostumasi perilaku robot. Hal itu dapat dilakukan dengan menggunakan fitur *built-in* yang disertakan sebagai fitur perangkat robot tersebut atau dengan membangun sebuah program dengan menggunakan bahasa pemrograman tertentu seperti C, C++, Java, dan bahasa pemrograman lainnya.

Java merupakan bahasa pemrograman berorientasi pada objek yang mengangkat objek-objek yang ada di dunia nyata dan memiliki fleksibilitas yang bagus sebagai salah satu bahasa pemrograman karena bersifat *multiplatform*, yang artinya dapat dijalankan pada berbagai *platform* seperti Linux, Windows, Solaris dan berbagai perangkat *mobile* tanpa harus mengubah kode yang sudah ada, yang biasa disebut dengan istilah "*write once run anywhere*". Hal ini dimungkinkan karena program java bisa berjalan dimana saja selama perangkat tersebut memiliki Java Virtual Machine (JVM) yang akan mengeksekusi program java nantinya.

Lego Mindstorms NXT merupakan sebuah perangkat robotik yang bisa diprogram dengan menggunakan aplikasi ROBOLAB yang merupakan bagian dari perangkat robot itu sendiri. Walaupun demikian, lego memberikan kemungkinan untuk mengganti *firmware* default lego dengan *firmware* yang dibuat secara terpisah. leJOS merupakan sebuah *firmware* yang memiliki fitur JVM sehingga memungkinkan *firmware* tersebut untuk menjalankan program java. Penggunaan Java sebagai bahasa pemrograman robot semakin meningkatkan eksplorasi di bidang robotik.

2 Tujuan

Berdasarkan latar belakang yang telah diuraikan sebelumnya penelitian dalam tugas akhir ini bertujuan untuk membuat sebuah library yang berisi gerakan primitif robot yang akan mempermudah eksplorasi pemrograman robot dengan Java. Selain itu akan peneliti juga bertujuan untuk memberikan gambaran bagaimana pemrograman robot menggunakan Java pada robot Lego Mindstorms NXT dengan menghasilkan gerakan primitif robot yang akan mampu menyelesaikan studi kasus line maze.

3 Studi Literatur

Bahasa pemrograman Java pertama kali dirilis pada tahun 1995. Pada awalnya digunakan untuk membangun sebuah proyek untuk Sun Microsystem (yang saat ini merupakan cabang dari perusahaan Oracle) dan merupakan bagian penting dari *Platform* Java Sun Microsystem [7]. Sintaks yang digunakan pada bahasa pemrograman Java diturunkan dari bahasa pemrograman C dan C++, namun bahasa pemrograman Java berorientasi pada pemodelan objek dan kelas sehingga lebih mudah dalam penggunaannya. Keuntungan

menggunakan Java antara lain bersifat *multiplatform*, berorientasi objek, memiliki library yang lengkap, dan fitur *automatic garbage collector*.

LEGO MINDSTORM NXT merupakan sebuah robot kit yang dapat diprogram dengan berbagai bahasa pemrograman seperti C dan Java setelah terlebih dahulu mengganti *firmware* lego dengan *firmware* yang sesuai dengan bahasa pemrograman yang akan digunakan. *Firmware* yang dapat digunakan sebagai pengganti antara lain leJOS NXJ, Robot C, BrickOS dan berbagai *firmware* lainnya. Perkembangan LEGO MINDSTORM NXT didorong oleh perkembangan RIS (*Robotic Invention System*) dalam mengembangkan perangkat robot yang memiliki *brick* yang dapat diprogram oleh *programmer*. *Brick* merupakan otak dari robot yang mengatur semua komponen yang ada pada robot. *Brick* yang terdapat pada LEGO MINDSTORM NXT memiliki beberapa fitur seperti 3 *motor port* (A,B, dan C), 4 *sensor port* (*ultrasonic, light, touch, sound*), USB *port*, dan *loudspeaker*.

Output port digunakan untuk motor penggerak berupa *servo motor* yang berfungsi untuk menggerakkan bagian-bagian robot. *Servo motor* yang disediakan memiliki sensor rotasi yang tertanam di dalamnya. Sensor ini memungkinkan kita untuk mengendalikan pergerakan robot secara tepat. Sensor rotasi mengukur rotasi motor dalam derajat atau rotasi (akurasi +/- 1 derajat). Satu rotasi sama dengan 360°, sehingga bila motor diatur untuk berputar 180°, maka sumbu motor akan berputar setengah putaran.

Sensor port digunakan untuk menerima input kondisi lingkungan robot dengan menggunakan beberapa sensor yang dimiliki oleh lego. Sensor cahaya adalah sensor pada robot untuk mengenali cahaya dan gelap. Sensor ini dapat mengukur intensitas cahaya pada suatu ruangan dan intensitas warna. Sensor cahaya dapat digunakan untuk membuat robot penjejak garis atau robot yang dapat memisahkan benda berdasarkan intensitas cahaya. Intensitas terang atau gelap warna diukur dengan persentasi tertentu. Semakin gelap maka intensitas warna semakin kecil. Sensor ultrasonik memungkinkan robot mendeteksi keberadaan objek. Sensor ini digunakan agar robot dapat menghindari penghalang, mengindera dan mengukur jarak, dan mendeteksi pergerakan. Sensor ultrasonik mengukur jarak dalam sentimeter dan inci, dan memiliki kemampuan untuk mengukur jarak 0-255 cm dengan ketelitian +/- 3 cm. Beberapa jenis sensor yang lain juga dimiliki oleh lego tetapi tidak dibahas dalam penelitian ini.

Seperti yang telah dijelaskan peniliti sebelumnya bahwa peneliti menggunakan leJOS sebagai *firmware* pengganti *firmware* lego. leJOS NXJ menyediakan fitur yang memungkinkan pemrograman *brick* LEGO MINDSTORM NXT menggunakan java kerena leJOS memiliki Java Virtual Machine (JVM). leJOS memiliki beberapa package antara lain: package inti java, package untuk mengatur komunikasi, package untuk mengatur *brick*, sensor dan penggerak pada NXT, dan *package* untuk navigasi robot lego.

Dalam penelitian ini dibuat program untuk menyelesaikan sebuah maze dengan menggunakan algoritma Backtrack. Algoritma Backtrack adalah algoritma pencarian berbasis DFS (*Depth First Search*). Sebagai bagian dari algoritma graf, pada algoritma ini dilakukan pencarian jalur yang tepat sampai pada titik akhir atau tujuan. Pencarian dilakukan dengan menelusuri semua jalur yang ada sampai menemukan titik tujuan. Algoritma ini dapat digambarkan seperti sebuah pohon yang memiliki jalur dari akar ke daun. Pencarian jalur yang paling sesuai pada pohon ini dilakukan dengan mencoba kemungkinan semua jalur yang ada. Ada kondisi dimana pencarian ini bermuara pada jalur buntu. Dimulai dari satu titik sebagai titik awal, pencarian dilakukan mengikuti jalur dari titik awal tersebut. Apabila jalur itu berakhir pada jalur yang buntu maka dilakukan *bakctrack* ke jalur yang berperan sebagai *parent* dan memilih jalur lain yang belum ditelusuri. Proses pencarian ini memungkinkan semua jalur akan ditelusuri. Pencarian ini dilakukan secara berulang-ulang sampai menemukan titik tujuan dan pencarian akan berhenti sampai pada titik tujuan. Pada algoritma ini, diterapkan aturan *left hand* atau *right hand*. Aturan ini adalah aturan proses pencarian jalur apakah selalu dimulai dengan menelusuri jalur sebelah kiri untuk *left hand* atau jalur sebelah kanan untuk *right hand*.

4 Permasalahan

Seperti yang diuraikan peneliti sebelumnya, dalam penelitian ini dilakukan pembangunan sebuah program java yang di-*deploy* pada perangkat robot lego yang telah menggunakan leJOS sebagai *firmware*-nya. Program yang dibangun berupa library java yang menyediakan API untuk memfungsikan robot lego dalam menyelesaikan permasalahan berupa *line maze* maupun permasalahan yang lain selama masih memenuhi constraint berupa bentuk robot dan garis yang membentuk *line maze*.

Perangkat robot LEGO MINDSTORM NXT yang digunakan terdiri dari bagian-bagian kecil yang terpisah satu dengan yang lain. Untuk memungkinkan robot bergerak mengikuti jalur pada *maze* maka robot harus dirangkai menjadi bentuk robot yang sesuai untuk menyelesaikan permasalahan *line maze*.

Jika dilihat secara umum, pergerakan yang dibutuhkan oleh sebuah robot dalam menyelesaikan sebuah line maze hanya berupa berberapa gerakan dasar yang selanjutnya dalam lingkup makalah ini akan disebut sebagai gerakan primitif. Gerakan primitif robot merupakan gerakan dasar robot seperti berjalan ke depan, ke belakang, ke kiri dan ke kanan. leJOS memiliki library yang menyediakan package navigation

yang berisi kelas TachoPilot. TachoPilot memiliki beberapa operasi untuk gerakan primitif robot seperti *forward*, *backward*, *travel*, dan *rotate*.

1. **Forward** merupakan gerakan motor berputar ke depan sehingga robot dapat berjalan maju ke depan.
2. **Backward** merupakan gerakan motor berputar ke belakang sehingga robot dapat berjalan mundur.
3. **Rotate** merupakan gerakan motor berputar beberapa derajat sehingga robot dapat berputar ke kanan atau ke kiri.
4. **Travel** merupakan gerakan motor berputar ke depan sesuai dengan jarak yang ditentukan.

Berdasarkan pengamatan terhadap gerakan primitif yang disediakan TachoPilot menunjukkan bahwa gerakan primitif tidak akurat 100%. Dalam implementasinya terjadi penyimpangan 5°-10° ketika robot melakukan *forward*, *backward*, dan *rotate*. Penyimpangan ini menyebabkan gerakan *forward* dan *backward* tidak tepat lurus, dan gerakan *rotate* tidak tepat sesuai dengan derajat yang ditentukan.

Line Maze sesuai dengan terjemahan secara gramatikal merupakan sebuah jalan atau simpang yang rumit karena terdiri dari banyak jalan yang yang saling berpotongan. *Line Maze* terdiri dari jalan yang biasanya ditandai dengan garis yang memiliki satu atau banyak persimpangan. Tidak semua jalan atau simpang yang terdapat pada *Line Maze* akan menuju pada titik akhir karena beberapa jalan atau garis pada *line maze* merupakan jalan buntu. Primitif yang dibuat sendiri dapat digunakan robot untuk menyelesaikan *line maze* ini. Permasalahan lebih lanjut dalam menyelesaikan *line maze* adalah bagaimana robot dapat menelusuri jalur *valid* diantara banyak jalur yang tidak *valid*. Disamping itu, permasalahan yang dihadapi adalah bagaimana robot mampu merekam penelusuran jalur yang *valid* demi mencapai efesiensi waktu untuk penelusuran maze pada putaran kedua.

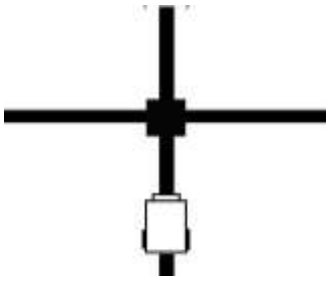
5 Penyelesaian Masalah

Untuk menyelesaikan permasalahan berupa line maze, Robot LEGO MINDSTORM NXT dirangkai menjadi sebuah robot line follower. yang terdiri dari bebepara komponen utama berupa; (a) *Brick* Lego sebagai komponen utama robot, (b) dua buah *servo motor* sebagai penggerak robot, (c) dua buah *light sensor* yang digunakan untuk membaca jalur *maze*, (d) sensor ultrasonik yang digunakan untuk mendeteksi bagian akhir atau titik tujuan dari *maze* berupa objek yang menghalangi pergerakan robot pada jalur *maze*. Bentuk robot lego yang dirangkai bisa dilihat pada gambar 1.

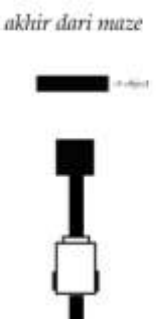
Gambar 1 Robot lego untuk maze solver

Gambar 2 Robot dengan sensor cahaya dan ultrasonik

Dua buah light sensor yang dirangkai sejajar – seperti yang digambarkan pada gambar 2 – digunakan untuk membaca jalur line maze dengan teknik menempatkan jalur hitam diantara kedua buah sensor tersebut. Hal dilakukan untuk mempercepat pembacaan jalur *maze*. Untuk mendefinisikan sebuah perpotongan atau belokan jalur maze, maka diidentifikasi dengan membuat kotak dengan ukuran melebih ukuran jarak sensor. Gambaran persimpangan jalur *maze* bisa dilihat pada gambar 2 dan gambaran ujung *maze* bisa dilihat pada gambar 3.



Gambar 3 Persimpangan maze



Gambar 4 Ujung maze dan akhir maze

Untuk mengatasi penyimpangan pada primitif API Lejos (TachoPilot), maka dibuat library yang menyediakan API gerakan primitif robot. Library yang dibuat menyediakan API untuk gerakan primitif robot berupa:

1. **Forward** merupakan gerakan motor berputar ke depan sehingga robot dapat berjalan maju ke depan.
2. **Backward** merupakan gerakan motor berputar ke belakang sehingga robot dapat berjalan mundur.
3. **Turnleft** merupakan gerakan robot berputar 90° ke kiri.
4. **Turnright** merupakan gerakan robot berputar 90° ke kanan.
5. **Turnleft** merupakan gerakan robot berputar 180°
6. **Travel** merupakan gerakan motor berputar ke depan sesuai dengan jarak yang ditentukan.

Pada library ini juga dibuat primitif untuk operasi sensor, antara lain:

1. **isWhite** merupakan operasi robot dapat mendeteksi benda yang memiliki permukaan putih menggunakan sensor cahaya.
2. **isBlack** merupakan operasi robot dapat mendeteksi benda yang memiliki permukaan hitam menggunakan sensor cahaya.
3. **isFoundObject** merupakan operasi robot dapat mendeteksi jarak benda menggunakan sensor ultrasonik.

Primitif yang dibuat sendiri dapat digunakan untuk menyelesaikan *line maze* sehingga robot dapat melalui jalur yang *valid*. Primitif yang dibuat dapat menghasilkan keakuratan yang lebih baik dari pada menggunakan primitif TachoPilot. Derajat penyimpangan *forward*, *backward*, *turnleft*, dan *turnright* dapat diperkecil sehingga hanya 1-3°. Penyimpangan ini dipengaruhi oleh faktor-faktor eksternal, seperti permukaan bidang gerak. Semakin halus permukaan bidang gerak maka penyimpangan yang terjadi semakin kecil.

Walaupun demikian primitif yang dibuat masih menimbulkan penyimpangan dalam proses melewati sebuah tikungan atau persimpangan jalur *maze*. Untuk mengatasi ini robot harus melakukan kalibrasi untuk memastikan robot berada pada posisi yang tepat sebelum melanjutkan penelusuran *maze* sehingga robot dapat mengikuti jalur yang valid. Proses kalibrasi ini dilakukan untuk memastikan robot tetap mengikuti garis dan ketika robot keluar dari garis 1-3°, dilakukan kalibrasi beberapa derajat ke kanan dan ke kiri, sampai robot dapat menemukan posisi yang sesuai pada garis. Kalibrasi ini dilakukan kira-kira 15° ke kiri dan kanan luar garis, sampai robot dapat menemukan garis kembali.

Untuk menemukan solusi *line maze* makan diterapkan algoritma backtrack. dalam hal ini dilakukan penerapan aturan *left hand rule* sehingga setiap gerakan robot pada *maze* selalu dimulai dari sebelah kiri, apabila garis tersebut menunjukkan simpul mati maka harus *backtrack* ke garis yang merupakan *parent*-nya dan menelusuri garis lain yang belum dilalui. Hal ini terus dilakukan sampai mencapai titik akhir yang ditandai oleh sebuah objek penghalang yang akan dideteksi menggunakan sensor ultrasonik.

Berdasarkan algoritma backtrack, tidak semua cabang dapat sampai pada titik akhir. Terdapat beberapa jalan buntu yang mengindikasikan jalur tersebut tidak *valid*. Apabila robot sampai pada jalur buntu maka robot akan melakukan gerakan primitif TurnBack, dan kembali ke *parent* dari garis.

Untuk meningkatkan efisiensi waktu penelusuran *maze* pada putaran kedua kalinya maka dilakukan penyimpanan setiap *state* dari jalur yang dilalui oleh robot pada percobaan pertama. Setiap jalur yang dilalui robot akan disimpan sebagai sebuah *state*, ditandai dengan simbol berupa huruf sesuai dengan arah yang dipilih robot antara lain:

1. **S** artinya *Straight*, robot bergerak Forward.
2. **L** artinya *Left*, robot bergerak TurnLeft.
3. **R** artinya *Right*, robot bergerak TurnRight.
4. **U** artinya *Turn*, robot menemukan jalur buntu dan melakukan gerakan primitif TurnBack.

Dengan adanya proses penyimpanan state ini maka akan dihasilkan sebuah string berupa solusi dari *maze* yang dibuat. Bentuk akhir dari string solusi tersebut beruapa rangkaian karakter S, L, dan R yang akan menggambarkan jalur dari titik awal ke titik tujuan maze.

6 Kesimpulan

Walaupun memiliki keterbatasan jika dibandingkan dengan bahasa pemrograman lainnya dalam lingkup *embedded programming*, pemrograman robot menggunakan Java relatif memberikan kemudahan dalam melakukan pembelajaran pemrograman robot. Ciri khas java sebagai bahasa pemrograman bersifat Object Oriented memudahkan peneliti dalam melakukan perancangan program yang dibangun. Selain itu, beberapa hal yang sebelumnya tidak bisa diimplementasikan menggunakan ROBOLAB, bisa diimplementasikan dengan menggunakan java (leJOS) dan *firmware* pengganti lainnya sehingga meningkatkan kemampuan robot LEGO MINDSORM NXT.

7 Saran

Penelusuran *maze* yang dilakukan robot menggunakan sensor cahaya dapat dibuat lebih mudah dengan memanfaatkan sensor warna sehingga titik cabang pada fisik *line maze* dapat dideteksi lebih mudah dengan perbedaam warna dari pada dengan memanfaatkan perbedaan intensitas cahaya.

8 Daftar Pustaka

[1] Juan Antonio Brena Moral, 2008, *Develop leJOS Programs Step by Step versi 0.4*

[2] Jason Butka, 2008, *Installing Lejos on the NXT Mindstorm using the Eclipse IDE and introduction to Lejos/ Java Progrqamming*
[3] ____,2010, *Intro to leJOS*, [online], http://lejos.sourceforge.net/nxt/nxj/tutorial/Preliminaries/Intro , diakses tanggal : 23 Feb 2010
[4] Adjie Nugroho, 2009, *Apa sih robot itu?*, [online], http://nugroho.staff.uii.ac.id/2009/02/01/apa-sih-robot-itu/ , diakses tanggal : 2 Maret 2010
[5] Guilo Ferrari, Andy Gombos, Soren Hilmer, Jurgen Stuber, Mick Porter, Jamie Waldinger dan Dario Laverde, 2002, *Programming Lego Mindstorms With Java*, Syngress, United States of America
[6] Eva Siahaan, Jogi Silalahi, Olivia Irma Sari, Pemrograman Robotik Menggunakan Bahasa Java, Tugas Akhir Diploma III Program Studi Teknik Informasi Politeknik Informatika Del, 2010
[7] Java programming languge, [online], http://en.wikipedia.org/wiki/Java_programming_language, , diakses tanggal : 15 Feb 2010