MODEL *MULTI-CLASS SVM* MENGGUNAKAN STRATEGI *1V1* UNTUK KLASIFIKASI *WALL-FOLLOWING ROBOT NAVIGATION DATA*

Azminuddin I. S. Azis[1], Vincent Suhartono[2], H. Himawan[3]
[123]Pascasarjana Teknik Informatika Universitas Dian Nuswantoro

ABSTRAK

Manusia memiliki keterbatasan dalam mengerjakan hal-hal yang berat, rumit, cepat, berbahaya, berulang-ulang secara langsung baik itu di kalangan rumah tangga, industri, militer, penelitian, hiburan, dsb. Robot merupakan mesin yang dapat mempermudah pekerjaan dan mengatasi keterbatasan manusia, sedangkan AI dapat membuat robot semakin cerdas. Berbagai macam metode AI telah diusulkan untuk mengatasi salah satu teknik navigasi robot, yaitu wall-following robot navigation, namun masih belum optimal. State of the art dalam klasifikasi wall-following robot navigation data adalah MLP dengan akurasi sebesar 97.59%. Namun akhir-akhir ini, state of the art dalam klasifikasi pattern recognition adalah SVM. Wall- following robot navigation data melibatkan multi-class, non-linear, dan high dimensional problem. 1V1 merupakan strategi terbaik yang dapat diterapkan pada SVM untuk mengatasi multi class problem yang selanjutnya dapat disebut multi-class SVM. Sedangkan untuk mengatasi non-linear dan high dimensional problem, SVM sendiri sudah dimodifikasi dengan memasukkan fungsi Kernel. Dengan demikian, akurasi sebesar 97.59% yang dihasilkan oleh MLP untuk klasifikasi wall-following robot navigation data masih dianggap rendah. Namun model multi-class SVM menggunakan strategi 1V1 untuk klasifikasi wall-following robot navigation data yang diperoleh dengan solusi yang global optimal dan tanpa perlu adanya dimensionality reduction (by PCA/SVD) dalam tingkat akurasi fair classification, yaitu 91.10% < 97.59% yang dihasilkan penelitian sebelumnya menggunakan MLP. Namun secara teoritis, multi-class SVM menggunakan strategi 1V1 lebih cepat dengan waktu proses yang dihasilkan = 10.7505 detik. Dengan demikian, model tersebut dapat mempelajari navigasi robot pengikut dinding tanpa tabrakan (menjaga jarak terhadap dinding dengan baik).

Kata Kunci: Multi-Class SVM, 1V1, Klasifikasi, Wall-Following Robot Navigation Data.

1. PENDAHULUAN

1.1. Latar Belakang

Artificial intelligence (AI) merupakan cabang dari *computer science* (ilmu komputer) yang dalam merepresentasikan pengetahuan lebih banyak menggunakan bentuk simbol-simbol daripada bilangan dan memproses informasi berdasarkan metode heuristik atau berdasarkan sejumlah aturan [1]. Dengan demikian, AI memperlakukan representasi pengetahuan dan manipulasinya, merepresentasikan inti dari ilmu komputer, dan mewujudkan suatu bentuk ketidak tepatan dari komputasi (karakteristik dalam matematika). Alan Turing, ahli matematika berkebangsaan Inggris yang dijuluki bapak komputer modern menetapkan definisi AI: "Jika komputer tidak dapat dibedakan dengan manusia saat berbincang melalui terminal komputer, maka bisa dikatakan komputer itu cerdas, mempunyai kecerdasan [1]." Sedangkan menurut Herbert A. S: "*Artificial intelligence* merupakan kawasan penelitian, aplikasi, dan instruksi yang terkait dengan pemrograman komputer untuk melakukan sesuatu hal yang dalam pandangan manusia adalah cerdas [1]." Karakter cerdas yang mulai dibutuhkan di berbagai disiplin ilmu dan teknologi membuat persoalan-persoalan yang ditangani oleh AI semakin berkembang, sehingga AI mempunyai suatu kekuatan alami antar cabang ilmu. Sebagai contoh, perpaduan antara teknik elektro dengan AI melahirkan *pattern recognition* [1].

Pattern recognition merupakan salah satu cabang dalam ilmu komputer yang cukup banyak diminati

dan terus mengalami perkembangan yang pesat. Hal ini dapat ditunjukkan dari berbagai penelitian-penelitian yang telah dilaksanakan. Contoh *Pattern recognition* sangat luas, beberapa diantaranya seperti: identifikasi biometrik (*face recognition, voice recognition, finger print recognition,* dll), diagnosis penyakit pasien secara otomatis, *complex letter recognition, line following robot navigation, wall following robot navigation,* dll.

Manusia memiliki keterbatasan dalam mengerjakan hal-hal yang berat, rumit, cepat, berbahaya, berulang-ulang secara langsung baik itu di kalangan rumah tangga, industri, militer, penelitian, hiburan, dsb. Robot merupakan mesin yang dapat mempermudah pekerjaan dan mengatasi keterbatasan manusisa, sedangkan AI dapat membuat robot semakin cerdas. Dengan demikian peningkatan kecerdasan robot merupakan hal yang penting. Saat ini, penggunaan kamera untuk navigasi robot (robot vision) memang sangat *trend*, terlebih untuk robot di udara ataupun di dalam air, namun bukan berarti sudah harus menggantikan peran sensor analog, sensor biner, laser, infrared, dll maupun berbagai teknik navigasi robot, seperti *line-following, wall-following,* dll. Sehingga biasanya berbagai alat dan teknik tersebut dipadukan pada suatu robot karena kelebihan dan kelemahannya masing-masing.

Pada awalnya teknik *wall-following robot navigation* menggunakan pemrograman konvensional yang hanya diperuntukkan sebagai alat hitung. Dengan hadirnya AI, maka penelitian tentang suatu metode AI untuk teknik *wall-following robot navigation* dapat menjadi topik yang menarik. Navigasi robot merupakan masalah yang paling penting dalam *mobile robot*, terutama di lingkungan yang tidak diketahuinya [2,3]. *Wall-following robot navigation* masih merupakan pekerjaan yang cukup rumit karena melibatkan pengolahan berkecepatan tinggi namun membutuhkan kapasitas memori yang sedikit serta membutuhkan akurasi yang tinggi dalam deteksi penghalang, pengukuran jarak, sudut, penanda, dll [4]. Berbagai macam metode AI telah diusulkan untuk mengatasi masalah ini, namun masih belum optimal [3]. Dengan demikian, efisiensi akurasi dan kompleksitas komputasi dengan menggunakan metode AI pada masalah *wall-following robot navigation* merupakan hal yang mutlak dibutuhkan. UCI Machine Learning Repository telah menyiapkan sampel data *wall-following robot navigation* yang dapat dimanfaatkan sebagai *dataset* pada penelitian ini.

Pattern recognition melakukan pemetaan suatu data ke dalam konsep tertentu yang telah didefenisikan sebelumnya [5]. Konsep tertentu yang dimaksud adalah *class* atau *category*. Sementara *wall-following robot navigation data* merupakan masalah yang dapat diklasifikasikan [6]. *Classification* (klasifikasi) merupakan konsep/teknologi yang cukup penting di dalam ilmu AI. Beberapa metode/algoritma yang umum dikenal dalam *classification* adalah: *Linear Discrimination Analysis* (LDA), *Hidden Markov Model*, *Artificial Neural Network* (ANN), *Fuzzy*, *Support Vector Machines* (SVM), *Bayesian*, *Boosting*, *Nearest Neighbor* (NN), *Naïve Bayes*, dll [7].

Pada tahun 2011, Yousefi Azar Khanian M *et al*. mengusulkan sebuah algoritma navigasi untuk *wall-following robot* menggunakan metode *Fuzzy Kalman Filter* dengan hasil simulasi yang menunjukkan bahwa algoritma yang diusulkan menawarkan keunggulan dan meningkatkan kinerja dibandingkan dengan metode-metode sebelumnya untuk masalah tersebut [3]. Namun penelitian ini hanya menyelesaikan *linear problem* dan tidak menggunakan *dataset wall-following robot navigation data* di UCI Machine Learning Repository yang merupakan *non-linear problem*. Sedangkan penelitian terakhir yang menggunakan *dataset* tersebut adalah penelitian yang dilakukan oleh Ananda L. Freire *et al*. pada tahun 2009. Penelitian tersebut menyelidiki pengaruh jangka pendek mekanisme memori pada kinerja klasifikasi menggunakan beberapa arsitektur standar *Neural Network,* yaitu: *Logistic Perceptron* (LP), *Multilayer Perceptron* (MLP), *Mixture of Experts* (ME), dan *Elman Network* (EN) yang dirumuskan sebagai masalah klasifikasi *pattern recognition* dimana pengklasifikasi *linear* seperti LP tidak dapat mempelajari tugas dan perintah pada *mobile robot* tanpa tabrakan, sedangkan klasifikasi *non-linear* seperti MLP, mampu mempelajari tugas dan perintah pada *mobile robot* tanpa tabrakan dengan akurasi sebesar 97.59%, selanjutnya klasifikasi dinamis EN dapat mempelajari tugas dan perintah pada *mobile robot* menggunakan *Less Hidden Neuron* dari MLP [6].

Namun salah satu metode yang akhir-akhir ini banyak mendapat perhatian sebagai *state of the art* dalam klasifikasi *pattern recognition* adalah SVM [8,9,10,11]. Selain itu, *future work* dari penelitin

terakhir pada *dataset wall-following robot navigation data* di UCI Machine Learning Repository mengusulkan metode SVM [6]. Usia SVM yang relatif terbilang masih muda secara teoritik dikembangkan untuk masalah *binary classification* (klasifikasi dua kelas) [9,12], seperti yang dikemukakan oleh Cortes *and* Vapnik, bahwa: "*Support Vector Machine (SVM) originally separates the binary classes (k=2) with a maximized margin criterion* [9]." Usaha menemukan *hyperplane* yang terbaik pada *input space* yang dapat bekerja pada masalah *non-linear* dengan cara memasukkan konsep *kerner trick* pada ruang kerja berdimensi tinggi, membuat SVM dewasa ini termasuk telah berhasil diaplikasikan pada *real-world problem* dan secara umum memberikan solusi yang lebih baik dibandingkan metode konvensional seperti misalnya ANN [5,11]. Pendekatan *Structural Risk Minimization* (SRM) pada SVM memberikan *error generalisasi* yang lebih kecil daripada yang diperoleh dari strategi *Empirical Risk Minimization* (ERM) pada ANN maupun metode yang lainnya, Vapnik mampu membuktikan bahwa SVM merupakan metode yang tepat untuk digunakan dalam memecahkan masalah berdimensi tinggi dan dari keterbatasan sampel data yang ada [5,11]. Beberapa kelebihan lainnya antara lain: SVM memiliki landasan teori yang dapat dianalisa dengan jelas dan tidak bersifat *black box*, dapat diselesaikan dengan metode sekuensial dan relatif mudah untuk diimplementasikan [5]. Namun di samping kelebihannya, SVM juga memiliki kelemahan atau keterbatasan, antara lain: sulit digunakan pada masalah berskala besar (jumlah sample yang diolah besar) ataupun keterbatasannya dalam menangani *multi-class problem* [5,9].

Jika pada metode lain, seperti ANN semua data latih akan dipelajari selama proses pelatihan, pada SVM berbeda, kerana hanya sejumlah data terpilih saja yang berkonstribusi untuk membentuk model yang akan digunakan dalam klasifikasi/prediksi yang akan dipelajari [11]. Hal ini menjadi kelebihan SVM, karena tidak semua data latih akan dipandang untuk dilibatkan dalam setiap iterasi pelatihannya. Dengan demikian SVM dianggap bisa lebih cepat daripada metode lainnya.

Penelitian lebih lanjut terus berusaha mengembangkan SVM sehingga bisa memecahkan masalah *multi-class* dengan baik [12], karena bagaimanapun *real-world problem* (dalam aplikasi praktis) cenderung melibatkan *multi-lass problem* [13]. Seperti yang diungkapkan oleh Mori *et al*., bahwa: "SVM pada awalnya hanya memisahkan dua *class* dengan cara mamaksimumkan kriteria *margin*. Namun pada masalah di dunia nyata sering membutuhkan klasifikasi yang lebih dari dua kelas [9]," salah satu contohnya adalah klasifikasi pada *wall-following robot navigation data*, karena *dataset* ini memiliki *class* yang lebih dari dua.

Ada dua pilihan untuk mengimplementasikan *multi-class* SVM, yaitu dengan menggabungkan beberapa SVM biner atau menggabungkan semua data yang terdiri dari beberapa *class* ke dalam bentuk optimasi [12]. Dalam prakteknya, masalah *multi-class* (k>2) umumnya didekomposisi menjadi serangkaian masalah biner sehingga standar SVM dapat langsung diterapkan [9]. Beberapa pendekatan/strategi yang dapat diterapkan untuk memecahkan masalah tersebut antara lain: *One-Versus-Rest* (1VR) oleh Vapnik pada tahun 1998 dan *One-Versus-One* (1V1) oleh Krebel pada tahun 1999. Dietterich and Bakiri pada tahun 1995 menyatakan bahwa keduanya bagus dalam mengatasi masalah *Error Correcting Output Codes* (ECOC) [9].

Pada penelitian lainnya yang dilakukan oleh Vapnik pada tahun 1998, Weston & Watkins (W&W) pada tahun 1999, Bredensteiner & Bennett pada tahun 1999, Guermeur pada tahun 2002, dan Crammer & Singer (C&S) pada tahun 2001 menangani masalah *multi-class* dalam satu proses optimasi tunggal. Mereka mengkombinasikan masalah *multiple binary-class optimization* menjadi satu fungsi tujuan tunggal dan secara bersamaan mencapai *multi-class problem*, namun berakibat kompleksitas komputasi yang besar karena ukuran pada masalah *Quadratic Programming* (QP) [12,9].

Selanjutnya, Szedmak *et al*. pada tahun 2004 mengusulkan sebuah model *multi-class*. Dalam formulanya, pendekatan 1VR diterapkan. Namun potensi masalah terjadi saat jumlah *class* terlalu besar sehingga terjadi *unbalanced* pada setiap klasifikasi biner. Masalah *unbalanced classification* hadir ketika terdapat banyak jumlah sampel yang sama dengan *class*. Sedangkan SVM sensitif terhadap *unbalanced classification* yang tinggi sejak karena menghasilkan klasifikasi yang memiliki estimasi bias yang kuat terhadap *class mayor* sehingga dapat memberikan akurasi yang buruk pada performasi akurasi pada *class minor*, yang mana hal ini didiskusikan pada beberapa penelitian yang dilakukan oleh Chawla *et al*. pada

tahun 2004 & Tang *et al* pada tahun 2009 [9].

Wang & Shen pada tahun 2007 mengusulkan sebuah metode dengan menggunakan 1VR pula. Sedangkan Suykens & Vandewalle pada tahun 1999 memperluas metode LS-SVM menjadi salah satu pilihan untuk *multi-class problem*, namun kelemahan dari LS-SVM adalah solusinya dibangun dari sebahagian besar data *training* dimana hal ini disebut sebagai masalah *non-sparsenes* [9]. Namun pada tahun 2008 Xia & Lee menghadirkan sebuah metode LS-SVM yang baru untuk menangani masalah *multi-class* dimana solusinya merupakan *sparse* dalam koefisien berat pada *support vector*. Sedangkan Vang & Mangasarian pada tahun 2005 mengikuti metode Proximal-SVM yang diterapkan pada masalah *multi-class*. Namun semua pendekatan tersebut tetap saja berkaitan erat dengan pendekatan 1VR [9]. Sementara sebelumnya, pada tahun 2001, Crammer & Singer (C&S) menyatakan bahwa eksperimen mereka mengindikasikan *state-of-the-art accuracy* pada masalah *multi-class* SVM dengan menerapkan pendekatan *Kernel-based* [14]. Namun bagaimanapun, jumlah variabel pada metode C&S mengakibatkan kompleksitas komputasi yang besar [12,9].

Pendekatan lain untuk *multi-class* SVM adalah *One-Versus-One* (1V1) atau dekomposisi berpasangan yang diperkenalkan oleh Knerr *et al*. Pada tahun 1990. Pendekatan ini mengevaluasi semua *classifiers* yang mungkin berpasangan dan dengan demikian menginduksi $k(k-1)/2$ *binary classification*. Setiap *classifier* diterapkan ke data uji yang akan memberikan satu nilai untuk *class* pemenang. Ukuran pengklasifikasi yang diciptakan oleh pendekatan 1V1 jauh lebih besar dibandingkan dengan pendekatan 1VR. Namun, ukuran QP di setiap *classifier* lebih kecil, yang memungkinkan untuk lebih cepat dalam pelatihan. Selain itu, dibandingkan dengan pendekatan 1VR, 1V1 yang lebih simetris [9]. Merujuk pula pada penelitian yang dilakukan oleh oleh Hsu C & Lin C., pada tahun 2002 yang memberikan komparasi berbagai metode untuk masalah *multi-class* SVM dan menyatakan bahwa 1V1 yang terbaik saat itu [12]. Masih di tahun yang sama, penelitian mereka yang lain menyatakan bahwa metode dekomposisi merupakan metode utama untuk memperbaiki SVM [15], sementara 1V1 menggunakan metode dekomposisi.

Dari berbagai penelitian yang telah dipaparkan di atas telah memberikan ransangan minat penelitian di bidang *pattern recognition* untuk menginvestigasi potensi kemampuan SVM secara teoritis maupun dari segi aplikasi. Beberapa strategi yang telah dipaparkan di atas masih memiliki kelemahan masing-masing. Namun berdasarkan berbagai penelitian tersebut, strategi yang lebih akurat dengan kompleksitas komputasi yang lebih cepat dalam menyelesaikan masalah *multi-class* SVM adalah 1V1 bila dibandingkan dengan ketiga strategi *multi-class* SVM lainya: 1VR, W&W, dan C&S.

Dengan demikian, solusi yang diusulkan dalam penelitian ini menggunakan strategi 1V1 yang akan diterapkan pada SVM untuk klasifikasi *wall-following robot navigation data*. Untuk itu, formula yang disulkan oleh Hsu C & Lin C, yaitu BSVM (http://mloss.org/software/view/62/) [16] diadopsi, begitupun dengan LiblarySVM (LIBSVM) dan dengan menggunakan *tool* Matlab untuk melakukan eksperimen, dimana pengukuran kinerja klasifikasinya dilakukan dengan *confusion matrix* sehingga menghasilkan nilai akurasi dan *error rate*. Karena mempertimbangkan bahwa bila suatu metode semakin kompleks komputasinya (terutama terhadap ruang) maka biasanya semakin akurat pula, begitupun sebaliknya [17], sehingga informasi kompleksitas komputasi yang diberikan dalam penelitian ini hanyalah terhadap waktu dalam detik, sedangkan informasi kompleksitasnya terhadap ruang tidak disajikan. Selanjutnya, berbagai informasi tersebut dibandingkan pula dengan informasi-informasi yang didapatkan dari penetian terakhir pada *wall-following robot navigation data*, yaitu penelitian yang dilakukan oleh Ananda L. Freire *et al*. pada tahun 2009, menggunakan beberapa arsitektur standar ANN, dimana MLP merupakan metode yang terbaik dalam penelitian tersebut dengan tingkat akurasi = 97.59% [6] sehingga diharapkan akurasi yang dihasilkan *multi-class* SVM menggunakan 1V1 > 97.59%.

Diharapkan penelitian ini dapat mengoptimalkannya penerapan metode AI pada teknik *wall-following robot navigation* untuk meningkatkan efisiensi akurasi dan kompleksitas komputasinya sehingga dapat meningkatkan kecerdasan robot pada teknik *wall-following robot navigation* guna mempermudah pekerjaan dan mengatasi keterbatasan manusia dalam mengerjakan hal-hal yang berat, rumit, cepat, berbahaya, berulang-ulang secara langsung baik itu di kalangan rumah tangga, industri, militer, penelitian,

hiburan, dsb. Selain itu, diharapkan pula penelitian ini dapat menghasilkan suatu model *multi-class* SVM menggunakan strategi 1V1 dalam mengklasifikasikan *wall-following robot navigation data* yang lebih akurat daripada MLP.

1.2. Rumusan Masalah

Berdasarkan latar belakang yang telah diuraikan, maka dalam penelitian ini dapat dirumuskan permasalahan sebagai berikut:

a. Manusia memiliki keterbatasan dalam mengerjakan hal-hal yang berat, rumit, cepat, berbahaya, berulang-ulang secara langsung baik itu di kalangan rumah tangga, industri, militer, penelitian, hiburan, dsb. Robot merupakan mesin yang dapat mempermudah pekerjaan dan mengatasi keterbatasan manusisa, sedangkan AI dapat membuat robot semakin cerdas. Dengan demikian peningkatan kecerdasan robot merupakan hal yang penting. Salah satu teknik navigasi robot, yaitu *wall-following robot navigation* masih merupakan pekerjaan yang cukup rumit karena melibatkan pengolahan berkecepatan tinggi namun membutuhkan kapasitas memori yang sedikit serta membutuhkan akurasi yang tinggi dalam deteksi penghalang, pengukuran jarak, sudut, penanda, dll. Berbagai macam metode AI telah diusulkan untuk mengatasi masalah ini, namun masih belum optimal. Dengan demikian, efisiensi akurasi dan kompleksitas komputasi dengan menggunakan metode AI pada masalah *wall-following robot navigation* merupakan hal yang mutlak dibutuhkan.
b. *State of the art* dalam klasifikasi *wall-following robot navigation data* adalah MLP dengan akurasi sebesar 97.59%. Namun akhir-akhir ini, *state of the art* dalam klasifikasi *pattern recognition* adalah SVM yang pada awalnya telah berhasil digunakan untuk *binary classification* dan pada data yang tidak dapat dipisahkan secara linear serta berdimensi tinggi, namun aplikasi praktis lebih cenderung melibatkan *multi class* dan *non-linear problem*, misalnya seperti klasifikasi *wall-following robot navigation data*. Sementara strategi 1V1 merupakan strategi terbaik yang dapat diterapkan pada SVM untuk mengatasi *multi class problem* yang selanjutnya dapat disebut *multi-class* SVM, namun belum ada model *multi class* SVM menggunakan strategi 1V1 untuk klasifikasi *wall-following robot navigation data*. Selain itu, berbeda dengan MLP dan metode klasifikasi secara umumnya, SVM tidak melibatkan semua data latih dalam setiap iterasi pelatihannya sehingga lebih cepat. SRM pada SVM memberikan *error generalisasi* yang lebih kecil dari pada yang diperoleh metode klasifikasi lainnya. Semua hal tersebut sejalan dengan *future work* dari penelitin terakhir pada *dataset wall-following robot navigation data* menggunakan MLP yang mengusulkan SVM. Dengan demikian, akurasi sebesar 97.59% yang dihasilkan oleh MLP untuk klasifikasi *wall-following robot navigation data* masih dianggap rendah.

1.3. Tujuan

Berdasarkan rumusan masalah yang telah diuraikan, maka tujuan penelitian ini adalah:

a. Dioptimalkannya penerapan metode AI pada teknik *wall-following robot navigation* untuk meningkatkan efisiensi akurasi dan kompleksitas komputasinya sehingga dapat meningkatkan kecerdasan robot pada *teknik wall-following robot navigation*.
b. Diperolehnya model *multi-class* SVM menggunakan strategi 1V1 untuk klasifikasi *wall-following robot navigation data* dengan akurasi yang lebih besar dari pada MLP (97.59%).

1.4. Manfaat

Diharapkan, dampak yang dapat diberikan apabila tujuan penelitian ini tercapai adalah:

a. Hasil penelitian ini dapat menjadi sumbangan pemikiran ataupun karya yang bisa digunakan sebagai bahan pertimbangan atau solusi dalam upaya meningkatkan kecerdasan robot pada teknik *wall-following robot navigation* guna mempermudah pekerjaan dan mengatasi keterbatasan manusia dalam mengerjakan hal-hal yang berat, rumit, cepat, berbahaya, berulang-ulang secara langsung baik itu di kalangan rumah tangga, industri, militer, penelitian, hiburan, dsb.

b. Hasil penelitian ini dapat memberikan masukan bagi perkembangan ilmu pengetahuan dan teknologi, khusunya pada bidang *pattern recognition* dan robotika berupa suatu model *multi class* SVM menggunakan strategi 1V1 dalam mengklasifikasikan *wall-following robot navigation data* yang lebih akurat.

2. TINJAUAN PUSTAKA

2.1. Penelitian Terkait

Berbagai penelitian terkait, baik tentang *wall-following robot navigation* dirangkum dalam bentuk tabel berikut ini.

Tabel 1. Penelitian

PENELITI (Th)	JUDUL / MASALAH	METODE	HASIL
Yousefi Azar Khanian M et al (2011).[3]	Robot Navigation Algorithm to Wall Following Using Fuzzy Kalman Filter	Fuzzy Kalman Filter	Linear Separable sehingga tidak dapat dijadikan acuan
PENELITI (Th)	JUDUL / MASALAH	METODE	HASIL
Ananda L. Freire et al (2009).[6]	Short-Term Memory Mechanisms in Neural Network Learning of Robot Navigation Tasks: A Case Study	ANN (LP, ME, MLP, Elman)	**Akurasi MLP = 97.59%**

2.2. Landasan Teori

2.2.1 *Pattern Recognition*, Klasifikasi, dan Model

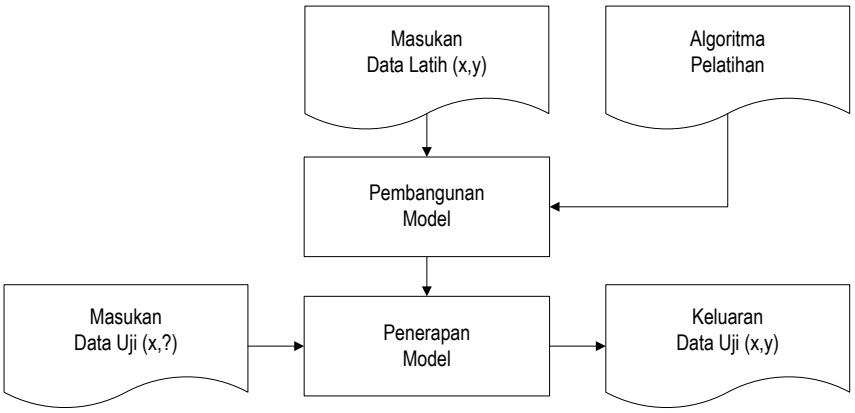
Menurut Hanif Al Fatha, "Pola adalah komposit/gabungan dari ciri yang merupakan sifat dari sebuah objek [21]." *Pattern recognition* (pengenalan pola) merupakan bagian dalam *machine learning*, dapat pula *computer vision* yang menurut Richard O. Duda *et al*. dapat diartikan sebagai: "tindakan mengambil data mentah dan bertindak berdasarkan klasifikasi data [7]." Dengan demikian, *pattern recognition* termasuk dalam *supervised learning*.

Pengenalan pola merupakan bagian dari AI yang menitikberatkan pada metode klasifikasi untuk menyelesaikan masalah tertentu. Beberapa contohnya adalah: *biometric identification*, *complex letter recognition*, *Optical Character Recognition* (OCR), *handwriting identification, robot vision, line/ wall following robot navigation,* dll. *Pattern recognition* bertujuan menentukan kategori pola berdasarkan ciri-ciri yang dimiliki oleh pola tersebut [23].

Pattern recognition melakukan pemetaan suatu data ke dalam konsep tertentu yang telah didefenisikan sebelumnya [5]. Konsep tertentu yang dimaksud adalah *class* atau *category*. Beberapa metode/algoritma yang umum dikenal dalam *classification* adalah: *Linear Discrimination Analysis* (LDA), *Hidden Markov Model*, *Artificial Neural Network* (ANN), *Fuzzy*, *Support Vector Machines* (SVM), *Bayesian, Boosting, Nearest Neighbor, Naïve Bayes,* dll [7].

Klasifikasi merupakan kata serapan dari bahasa Belanda, *classificatie*, yang sendirinya berasal dari bahasa Prancis *classification*. Istilah ini menunjuk kepada sebuah metode untuk menyusun data secara sistematis atau menurut beberapa aturan atau kaidah yang telah ditetapkan. Secara harafiah bisa pula dikatakan bahwa klasifikasi adalah pembagian sesuatu menurut kelas-kelas [24]. Dalam klasifikasi, ada 2 pekerjaan utama yang dilakukan, yaitu: (1) Pembangunan model sebagai prototipe untuk disimpan sebagai memori; (2) Penggunaan model tersebut untuk melakukan pengenalan/klasifikasi/prediksi pada suatu objek data lain agar diketahui di kelas mana objek data tersebut dalam model yang sudah disimpannya [11].

Model dalam klasifikasi mempunyai arti yang sama dengan kotak hitam, di mana ada suatu model yang menerima masukan, kemudian mampu melakukan pemikiran terhadap masukan tersebut, dan memberikan jawaban sebagai keluaran dari hasil pemikirannya [11]. *Framework* klasifikasi ditunjukkan pada gambar di bawah ini. Pada gambar tersebut disediakan sejumlah data latih (x,y) untuk digunakan sebagai data pembangunan model. Model tersebut kemudian digunakan untuk memprediksi kelas dari data uji (x,?) sehingga diketahui kelas *y* yang sesungguhnya.



Gambar 1. Proses Pekerjaan Klasifikasi [Sumber: 11]

Binary classification (klasifikasi biner atau binomial) adalah tugas mengelompokkan anggota himpunan benda menjadi dua kelompok/*class* (*k=2*) berdasarkan apakah mereka memiliki beberapa properti atau tidak [25]. Contohnya, tes medis untuk menentukan apakah pasien memiliki penyakit tertentu atau tidak (properti klasifikasi adalah adanya penyakit), apakah berjalan ke depan atau tidak, dsb. Selanjutnya, jika *binary classification* adalah *k=2*, maka *multi classification* adalah *k>2* [9].

Umumnya, pengukuran kinerja klasifikasi dilakukan dengan *confusion matrix* (matriks konfusi). *Confusion matrix* merupakan tabel pencatat hasil kerja klasifikasi. Tabel berikut ini menunjukkan contoh *confusion matrix*. Setiap sel f_{ij} dalam matriks menyatakan jumlah *record*/data dari kelas *i* yang hasil prediksinya masuk ke kelas *j* [11]. Misalnya, sel f_{11} adalah jumlah data dalam kelas 1 yang secara benar dipetakan ke kelas 1, dan f_{10} adalah data dalam kelas 1 yang dipetakan secara salah ke kelas 0.

Tabel 2. Contoh *Confusion Matrix* [11]

f_{ij}		Kelas Hasil Prediksi (*j*)	
		Kelas = 1	Kelas = 0
Kelas asli (*i*)	Kelas = 1	f_{11}	f_{10}
	Kelas = 0	f_{01}	f_{00}

Berdasarkan isi *confusion matrix*, maka dapat diketahui jumlah data dari masing-masing kelas yang diprediksi secara benar ($f_{11}+f_{00}$) dan data yang diklasifikasikan secara salah ($f_{10}+f_{01}$). Kuantitas *confusion matrix* dapat diringkas menjadi dua nilai, yaitu akurasi dan laju *error*. Untuk menghitung akurasi dan laju *error* (kesalahan prediksi) digunakan formula [11]:

$$\textbf{Akurasi} = \frac{\textbf{Jumlah data yang diprediksi secara benar}}{\textbf{Jumlah prediksi yang dilakukan}} \quad \text{.....(2.1)}$$

$$\textbf{Laju Error} = \frac{\textbf{Jumlah data yang diprediksi secara salah}}{\textbf{Jumlah prediksi yang dilakukan}} \quad \text{.....(2.2)}$$

2.2.2 *Dimensionality Reduction* dengan PCA dan SVD

Metode *dimensionality reduction* bekerja dengan cara tertentu untuk menangkap karakteristik data dengan memetakan *dataset* dari dimensi semula ke dimensi lain yang relatif rendah. Metode yang sudah digunakan secara luas untuk menyelesaikan masalah ini adalah: *Principal Component Analysis* (PCA) dan *Singular Value Decomposition* (SVD) [11]. PCA melakukan *dimensionality reduction* dengan memanfaatkan teknik dalam aljabar linear.

$$C_Y = PC_X P^T \quad (2.3)$$

Setiap beris matriks *P* adalah *eigenvector* C_X.

Jika PCA umumnya menggunakan *eigenvalue* dan *eigenvector* untuk mendapatkan solusi, SVD menggunakan dekomposisi nilai tunggal untuk mendapatkan solusi. Matriks X dapat dibentuk dengan persamaan berikut [11]:

$$A = \sum_{i=1}^{rank(A)} \sigma_i \mu_i v_i^T = U \sum V^T \quad (2.4)$$

σ_I adalah nilai singular ke-*i* dari nilai A (nilai ke-i pada diagonal $\sum$), μ_i adalah vektor singular kiri dari A (kolom ke-*i* dari U), dan v_i adalah vektor singular kanan ke-*i* dari A (kolom ke-*i* dari V).

2.2.3 SVM dan *One-Versus-One*

Dalam *machine learning*, *Support Vector Machine* (SVM) merupakan model *supervised learning* dalam menganalisis data dan *pattern recognition* yang digunakan untuk analisis klasifikasi dan regresi [26]. SVM pertama kali diperkenalkan oleh Vapnik bersama Boser & Guyon yang pertama kali dipresentasikan pada tahun 1992 di Annual Workshop on Computational Learning Theory. Evaluasi kemampuannya dalam berbagai aplikasinya menempatkannya sebagai *state of the art* dalam *pattern recognition*, dan dewasa ini merupakan salah satu tema yang berkembang dengan pesat [8,9,10,11]. SVM merupakan metode yang bekerja atas prinsip *Structural Risk Minimization* (SRM) dengan tujuan menemukan *hyperplane* terbaik yang memisahkan dua buah *class* pada *input space*, seperti yang dikemukakan oleh Cortes and Vapnik, bahwa: "*Support Vector Machine (SVM) originally separates the binary classes (k=2) with a maximized margin criterion* [9]."

Setiap data latih dinyatakan oleh (x_i, y_i), di mana *i = 1, 2, 3, …, N,* sehingga N adalah banyaknya data. Data yang tersedia dinotasikan sebagai: $\vec{x}_i \in R^d$, dimana $x_i = \{x_{i1}, x_{i2}, x_{i3}, \ldots, x_{iq}\}^T$ merupakan atribut (fitur) untuk data latih ke-*i*. Sedangkan label/kelas masing-masing dinotasikan sebagai: $y_i \in \{-1, +1\}$ di mana *i = 1, 2, 3, …,N*. Diasumsikan kedua kelas -1 dan +1 dapat terpisah secara sempurna oleh *hyperplane* berdimensi *d*, yang didefenisikan [5,11]:

$$\boldsymbol{w}.\boldsymbol{x_i} + \boldsymbol{b} = \boldsymbol{0} \quad (2.5)$$

Keterangan: $\boldsymbol{w}$ dan $\boldsymbol{b}$ merupakan parameter model yang ingin dicari nilainya. $\boldsymbol{w}.\boldsymbol{x_i}$ merupakan *inner-product* dalam antara $\boldsymbol{w}$ dan $\boldsymbol{x_i}$. *Data/pattern* $\boldsymbol{x_i}$ yang masuk *class* –1 dapat dirumuskan sebagai *pattern* yang memenuhi pertidaksamaan [5,11]:

$$\boldsymbol{w}.\boldsymbol{x_i} + \boldsymbol{b} \leq -\boldsymbol{1} \quad (2.6)$$

$$\boldsymbol{w}.\boldsymbol{x_i} + \boldsymbol{b} \geq +\boldsymbol{1} \quad (2.7)$$

Klasifikasi SVM pada Persamaan 2.6 dan 2.7 dapat digabungkan dengan notasi: $\boldsymbol{y_i(w.x_i + b) \geq 1, i = 1, 2, \ldots, N}$. *Margin* terbesar/optimal dapat ditemukan dengan memaksimalkan nilai jarak antara *hyperplane* dan titik/data terdekatnya. Selanjutnya, masalah ini diformulasikan ke dalam (dapat dirumuskan sebagai) *quadratic programming (QP) problem* [5,11]:

$$\min_{\vec{w}} \tau(\boldsymbol{w}) = \frac{1}{2}\|\vec{\boldsymbol{w}}\|^2 \quad (2.8)$$

Syarat:

$$\boldsymbol{y_i(w.x_i + b) - 1 \geq 0}$$

$$\boldsymbol{y_i(w.x_i + b) \geq 1, i = 1, 2, \ldots, N} \quad (2.9)$$

Optimalisasi ini dapat diselesaikan dengan *Lagrange Multiplier* [11]:

$$\boldsymbol{Lp = \frac{1}{2}\|w\|^2 - \sum_{i=1}^{N} \alpha_i y_i (w.x_i + b) - 1} \quad (2.10)$$

Penjelasan di atas berdasarkan asumsi bahwa kedua belah *class* dapat terpisah secara sempurna oleh

hyperplane. Akan tetapi umumnya dua buah *class* pada *input space* tidak dapat terpisah secara sempurna. Hal ini menyebabkan proses optimalisasi tidak dapat diselesaikan karena tidak ada *w* dan *b* yang memenuhi Pertidaksamaan 2.9. Dengan begitu, pertidaksamaan tersebut dimodifikasi dengan memperkenalkan teknik *soft margin*. Dalam *soft margin*, pertidaksamaan tersebut dimodifikasi dengan memasukkan variabel *slack* $\xi_i (\xi_i \geq 0)$ demikian juga dengan persamaannya (2.8) [9,11,28].

$$\underset{w \in R, b \in R, \xi \in R}{min} = \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{N}\xi_i \quad \text{.....(2.11)}$$

Syarat:

$$y_i(w.x_i + b) \geq 1 - \xi_i \quad \text{.....(2.12)}$$

Keterangan: $w \in R^d$ adalah *weight vector*, $C \in R$ untuk mengontrol *trade off* antara *margin* dan *error* klasifikasi *slack*. Parameter C berguna untuk mengontrol pertukaran antara *margin* dan *error* klasifikasi. Nilai *C* yang besar merupakan pinalti yang lebih besar terhadap *error* klasifikasi tersebut.

Jika dalam ANN ada *Perceptron* dan *Multi Layer Perceptron* (MLP), dalam SVM ada SVM *linear* dan SVM *non-linear* (*Kernel Trick*). Seperti halnya *Perceptron*, SVM sebenarnya adalah *hyperplane linear* yang hanya bekerja pada data yang dapat dipisahkan secara linear. Untuk data yang distribusi kelasnya tidak linear, maka digunakan pendekatan *Kernel* pada fitur data awal *dataset*. *C* pada Persamaan 2.11 merupakan konstanta regularisasi dan fungsi pemetaan φ data pelatihan ke *feature space* yang cocok. Sehingga memungkinkan untuk menyelesaikan masalah *non-linear* [9, 11]. Fungsi *Kernel* yang biasanya digunakan pada SVM, yaitu [5,11,27]:

1. *Linear*:

   $$K(x_i, x_j) = x_i.x_j \quad \text{.....(2.13)}$$

2. *Polynomial*:

   $$K(x_i, x_j) = (x_i.x_j + 1)^d \quad \text{.....(2.14)}$$

3. *Radial Basis Function (RBF)/Gaussian*:

   $$K(x_i, x_j) = \exp\left(\frac{\|x_i.x_j\|^2}{2\sigma^2}\right) \quad \text{.....(2.15)}$$

4. *Tangent Hyperbolic/Sigmoid*:

   $$K(x_i, x_j) = \tanh(\sigma(x_i.x_j) + c) \quad \text{.....(2.16)}$$

5. *Invers Multikuadrik*:

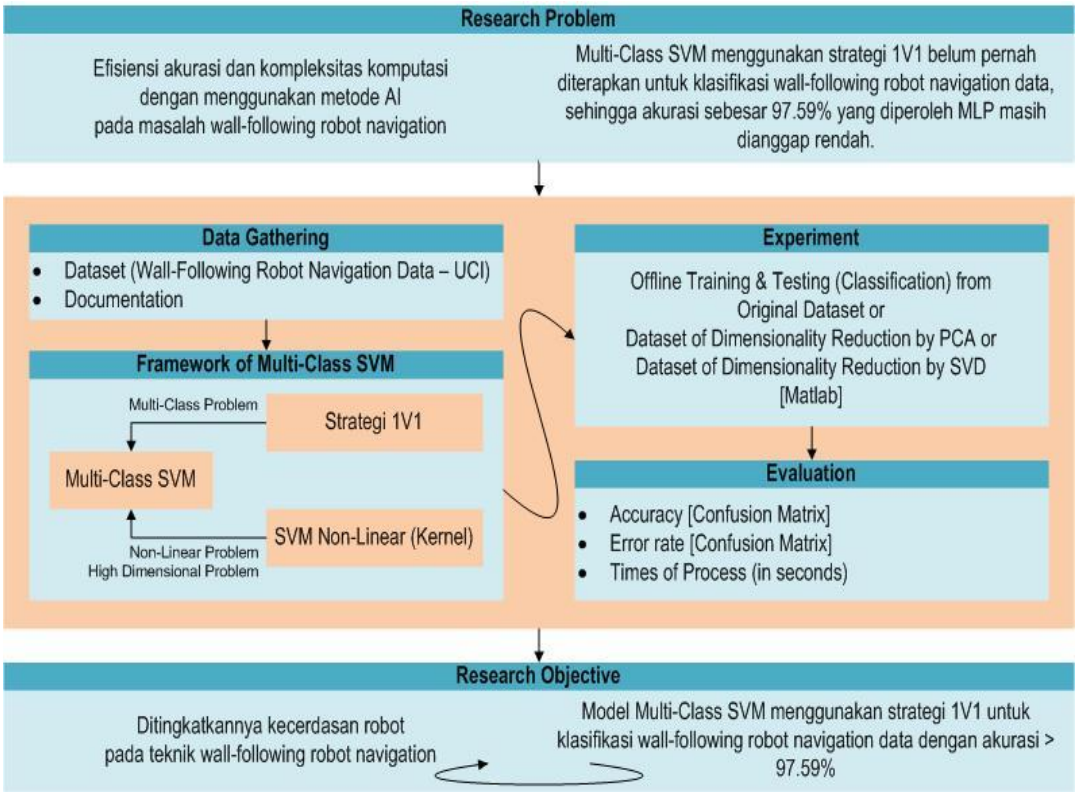
   $$K(x_i, x_j) = \frac{1}{\sqrt{\|x_i - x_j\|^2 + c^2}} \quad \text{.....(2.17)}$$

   σ, *c*, *d* > 0, merupakan konstanta.

Pendekatan/strategi *One-Versus-One* (1V1) atau dekomposisi berpasangan diperkenalkan oleh Knerr *et al.* pada tahun 1990. Pendekatan ini mengevaluasi semua *classifiers* yang mungkin berpasangan dan dengan demikian menginduksi $k(k-1)/2$ *binary classification*. Setiap *classifier* diterapkan ke data uji yang akan memberikan satu nilai untuk *class* pemenang [9]

3. METODE PENELITIAN

Berikut ini adalah gambaran singkat atau diagram alir tentang metode yang digunakan dalam penelitian ini.



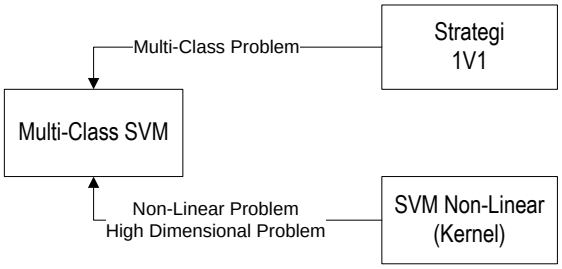
Gambar 2.3: Diagram Alir Metode Penelitian

3.1. *Data Gathering*

Data primer dalam penelitian ini adalah sebuah *dataset wall-following robot navigation data* yang dikumpulkan dari UCI Machine Learning Repository. Komposisi data latih dan data uji dipilih secara acak dengan teknik *holdout*. Terdiri dari 24 atribut di mana tiap-tiap atribut merupakan nilai dari pembacaan sensor ultrasound yang ditelakkan pada robot. *Class* terdiri dari 4 (*Move-Forward, Slight-Right-Turn, Sharp-Right-Turn,* dan *Slight-Left-Turn*). Jumlah data = 5456. Sedangkan instrumen yang digunakan untuk mengumpulan data sekunder adalah studi dokumentasi, yaitu studi terhadap buku, *paper*, jurnal, serta dokumen-dokumen yang terkait.

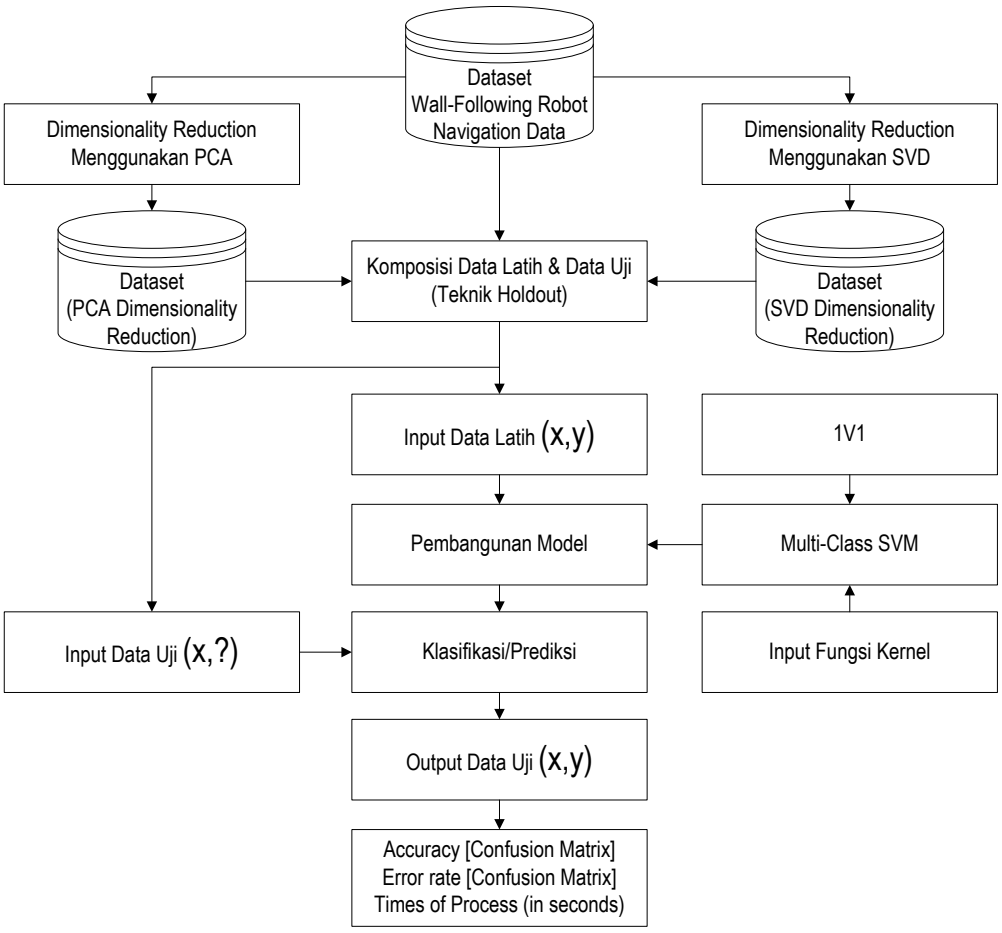
3.2. *Framework of Experiment*

Kerangka kerja *multi-class* SVM dapat ditunjukkan pada gambar berikut ini.



Gambar 2. *Framework of Multi-Class* SVM

Sedangkan kerangka eksperimen dapat ditunjukkan pada gambar berikut.



Gambar 3. *Framework of Experiment*

3.3. *Experiment*

Sebelum *dataset* dilatih dan diuji, terlebih dahulu dilakukan *data pre-processing* menggunakan teknik *dimensionality reduction* dengan menggunakan algoritma PCA dan SVD terhadap *dataset wall-following robot navigation data*. Dengan demikian, terdapat tiga dataset yang akan dilatih dan diuji menggunakan *tool* Matlab, yaitu: (1) *original dataset*; (2) *dataset* dari hasil *dimensionality reduction* menggunakan PCA; (3) *dataset* dari hasil *dimensionality reduction* menggunakan SVD.

Karena distribusi kelas pada *dataset* tersebut tidak dapat dipisahkan secara linear, maka digunakan metode SVM *non-linear* dengan memasukkan fungsi *Kernel* pada pelatihannya. Selain itu, karena *dataset* tersebut merupakan masalah *multi-class*, maka strategi 1V1 digunakan agar SVM mampu menangani masalah ini yang selanjutnya dapat disebut sebagai *multi-class* SVM. Strategi 1V1 cukup sederhana, yaitu hanya membagi semua kelas dari *dataset* yang mungkin berpasangan dengan menginduksi $k(k-1)/2$ *binary classification*.

3.4. *Evaluation*

Untuk mengukur kinerja klasifikasinya, evaluasi dilakukan menggunakan teknik *confusion matrix* dengan memanfaatkan *tool* Matlab sehingga didapatkan nilai akurasinya dan *error rate*-nya. Selain itu, nilai kompleksitas komputasi terhadap waktu disajikan pula**.** Semua hasil tersebut akan dibahas dan dibandingkan dengan penelitian terakhir pada *dataset* tersebut yang menggunakan MLP.

4. HASIL PENELITIAN DAN PEMBAHASAN

4.1. Hasil Penelitian

4.1.1 Opsi Eksperimen

Eksperimen penelitian ini terdiri dari beberapa opsi yang dapat ditunjukkan pada tabel berikut.

Tabel 3. Opsi Eksperimen

Opsi	Jumlah Data Uji	Fungsi Kernel	Dataset
1	**1090**	**polynomial (polyorder = 3)**	**Original**
2	sda	sda	Reduksi Dimensi PCA
3	sda	sda	Reduksi Dimensi SVD
4	sda	polynomial (polyorder = 8)	Original
5	sda	Sda	Reduksi Dimensi PCA
6	sda	Sda	Reduksi Dimensi SVD
7	sda	rbf (rbf_sigma = 1)	Original
8	sda	Sda	Reduksi Dimensi PCA
9	sda	Sda	Reduksi Dimensi SVD
10	4090	polynomial (polyorder = 3)	Original
11	sda	Sda	Reduksi Dimensi PCA
12	sda	Sda	Reduksi Dimensi SVD
13	sda	polynomial (polyorder = 8)	Original
14	sda	Sda	Reduksi Dimensi PCA
15	sda	Sda	Reduksi Dimensi SVD
16	sda	rbf (rbf_sigma = 1)	Original
17	sda	sda	Reduksi Dimensi PCA
18	sda	sda	Reduksi Dimensi SVD

Diantara 18 opsi tersebut, kinerja yang paling baik didapatkan dari opsi No 1 berdarkan evaluasi model dari setiap opsi tersebut yang akan dijelaskan dan dibuktikan pada sub bab Evaluasi Model.

4.1.2 *Dimensionality Reduction*

Hasil dari *dimensionality reduction* dapat ditunjukkan pada tabel-tabel berikut.

Tabel 4. Hasil *Dimensionality Reduction*

Nilai Eigenvalue dari *Dimensionality reduction* Menggunakan PCA					
US1 = 7.69	US2 = 5.64	US3 = 3.26	US4 = 2.98	US5 = 2.19	US6 = 1.97
US7 = 1.64	US8 = 1.50	US9 = 1.32	US10 = 1.13	US11 = 1.03	US12 = 0.97
US13 = 0.92	US14 = 0.81	US15 = 0.80	US16 = 0.72	US17 = 0.68	US18 = 0.65
US19 = 0.59	US20 = 0.53	US21 = 0.52	US22 = 0.43	US23 = 0.38	US24 = 0.37
Nilai Varian dari *Dimensionality reduction* Menggunakan SVD					
US1 = 1824.60	US2 = 1337.00	US3 = 773.90	US4 = 707.05	US5 = 521.22	US6 = 468.33
US7 = 388.14	US8 = 356.31	US9 = 313.43	US10 = 267.90	US11 = 243.94	US12 = 230.69
US13 = 219.57	US14 = 191.55	US15 = 191.05	US16 = 170.90	US17 = 162.64	US18 = 153.22
US19 = 141.18	US20 = 25.86	US21 = 123.43	US22 = 101.83	US23 = 89.34	US24 = 88.80

Berdasarkan pengamatan pada tabel di atas, semua atribut berpengaruh, namun agar berbeda dengan *original dataset*, maka hasil dari *dimensionality reduction* menggunakan PCA membuang 2 atribut terakhir (US23 dan US24), sedangkan hasil dari SVD membuang atribut terakhir (US24).

4.1.3 Komposisi Data Latih dan Data Uji

Kelas pada *dataset* bertipe data nominal, maka diubah ke numerik dengan memanfaatkan fungsi *grp2idx()* pada Matlab. Fungsi ini memiliki dua *output*, yang satu berisi setiap kelas dari setiap data yang diubah ke numerik, sedangkan *ouput* yang satunya lagi berisi 4 buah kelas saja (kelas duplikat dibuang) yang akan berguna untuk penerapan strategi 1V1, namun output yang ini bukan bertipe numerik, tetapi tetap aslinya. Dengan demikian, kelas *Slight-Right-Turn = 1; Sharp-Right-Turn = 2; Move-Forward = 3;* dan *Slight-Left-Turn = 4*. Selanjutnya, komposisi antara data latih dan data uji menggunakan teknik *'holdOut'* dengan memanfaatkan fungsi *crossvalind()* pada Matlab. Outputnya terdiri dari dua, yang satu berisi index kelas untuk data latih, sedangkan yang satunya lagi berisi index kelas untuk data uji.

4.1.4 Penerapan Strategi 1V1 dan Fungsi Kernel pada SVM

Dengan memanfaatkan fungsi *nchoosek()* pada Matlab, maka dekomposisi 4 kelas menjadi 6 kelas biner dapat ditunjukkan pada tabel berikut.

Tabel 5. Dekomposisi 4 Kelas ke 6 Kelas Biner

4(4-1)/2 = 6	Kelas +1	Kelas -1
Kelas	1	2
Kelas	1	3
Kelas	1	4
Kelas	2	3
Kelas	2	4
Kelas	3	4

Untuk menentukan *score* tertinggi dari ke-6 kelas biner tersebut, maka digunakan fungsi *mode()* pada Matlab. Fungsi ini mencari kelas pemenang dari 6 kelas yang ada dengan cara mencari kelas (antara 4 kelas) yang memiliki skor tertinggi dari 6 kelas biner tersebut. Fungsi *Kernel* yang digunakan adalah *'polynomial'* dengan *'polyorder'* = 3 (default value) dan fungsi kernel RBF.

4.1.5 Pelatihan dan Pengujian Model

Hasil pelatihan dan pengujian *model multi-class* SVM menggunakan strategi 1V1 untuk klasifikasi *wall-following robot navigation data* dapat ditunjukkan pada tabel-tabel berikut.

Tabel 6. Hasil Pelatihan Model (Opsi No.1)

	1V2		1V3		1V4		2V3		2V4		3V4	
Min/Max	Min	Max	Min	Max	Min	Max	Min	Max	Min	Max	Min	Max
SV	-1.6095	4.8435	-1.8587	6.0551	-1.8962	6.3837	-1.6686	5.9444	-1.5159	4.5182	-1.5233	6.6218
Alpha	-0.0937	0.1156	-0.1375	0.2101	-0.0029	0.0013	-0.1951	0.2051	-0.0092	0.0048	-0.0158	0.0156
Bias	-0.1464	-0.1464	-1.765	-1.765	-0.2678	-0.2678	-1.8886	-1.8886	0.045	0.045	0.6898	0.6898
SV Indices	6	2310	6	1415	10	877	10	3438	16	1939	4	2018

Tabel 7. Hasil Pengujian/Klasifikasi/Prediksi Model (Opsi No.1)

Kelas Uji Asli	Kelas Hasil (6 Kelas Biner)						Kelas Hasil	Benar Salah
	1V2	1V3	1V4	2V3	2V4	3V4		
1	1	1	1	2	2	3	1	Benar
1	2	3	1	2	2	3	2	Salah
…	…	…	…	…	…	…	…	…
2	2	3	4	2	2	3	2	Benar
							Hasil benar	**993**
							Hasil salah	**97**

4.1.6 Evaluasi dan Analisis Model

Hasil evaluasi *model multi-class* SVM menggunakan strategi 1V1 untuk klasifikasi *wall-following robot navigation data* dari opsi No.1 (yang terbaik) dibandingkan dengan penelitian terakhir pada *dataset* dalam penelitian ini yang menggunakan MLP sebagai metode yang terbaik saat itu.

Tabel 8. Komparasi Hasil Evaluasi Multi-Class SVM vs MLP

Method	Multi-Class SVM				MLP			
Class	1	2	3	4	1	2	3	4
1	157	2	6	0	571	7	6	0
2	10	380	25	4	0	213	8	0
3	20	23	394	4	1	0	559	0
4	0	1	2	62	11	0	2	78
Accuracy	91.10%				**97.59%**			
Error rate	8.90%				**2.41%**			
Times	10.75 seconds							

4.2. Pembahasan

Hasil penelitian ini menunjukkan bahwa:

a. Model yang dihasilkan dari *original dataset* lebih akurat dibandingkan dari *dataset* yang direduksi dimensinya. *Multi-class* SVM menggunakan strategi 1V1 memang terkadang membutuhkan seleksi atribut (reduksi dimensi) untuk meningkatkan kinerjanya, namun dapat pula sebaliknya. Secara teoritis, hal ini karena fungsi *Kernel* yang diterapkan pada SVM untuk mengatasi *non-linear problem* justru mentransformasikan data ke dimensi yang lebih tinggi, karena jika ditransformasikan ke ruang dimensi yang lebih rendah maka bisa beresiko menghilangkan *critical information* yang penting untuk mengklasifikasikan data. Hal ini sejalan dengan atribut/dimensi yang dimiliki pada *dataset wall-following robot navigation data,* di mana semua atributnya memang merupakan *critical information.*

b. Waktu proses yang dihasilkan dari *dataset* yang direduksi dimensinya menggunakan PCA paling cepat namun tidak akurat, sedangkan selisih waktu prosesnya tidak jauh berbeda dengan yang dihasilkan dari *original dataset*. Dengan demikian, hal ini tidak begitu memberikan pengaruh terhadap kinerja *multi-class* SVM dalam mengklasifikasikan *wall-following robot navigation data*. Karena cepat namun tidak akurat bukanlah hasil yang diharapkan dalam penelitian ini.

c. Komposisi data uji sangat mempengaruhi hasil akurasi dan waktu prosesnya di mana bila semakin banyak data uji maka semakin buruk akurasinya namun semakin cepat waktu prosesnya. Dalam penelitian ini, komposisi tersebut dilakukan secara acak menggunakan teknik *'holdout'* pada metode *cross validation* dengan parameter 'P' = 0.2 untuk data uji = 1090 dan 'P' = 0.9 untuk data

uji = 4909. Dengan demikian, komposisi data uji yang lain dan tepat sangat dibutuhkan untuk meningkatkan kinerjanya.

d. Pemilihan fungsi *Kernel* yang tepat sangat mempengaruhi hasil akurasi. Karena fungsi *Kernel* ini akan menentukan *feature space* di mana fungsi *classifier* (hyperplane) akan dicari. Sepanjang fungsi *Kernel*-nya sah, SVM akan beroperasi secara benar, meskipun kita tidak tahu seperti apa pemetaan yang dilakukan. Menurut Hastie et al. (2001) Biasanya metoda *cross-validation* digunakan untuk pemilihan fungsi kernel ini. Dalam penelitian ini, fungsi *Kernel* yang digunakan adalah *'polynomial'* dengan *value* '*polyorder'* = 3 (default) dan 'RBF'. untuk itu, dibutuhkan pemilihan nilai *polyorder* yg tepat apabila menggunakan fungsi *Kernel 'polynomial'* maupun percobaan pula dengan menggunakan fungsi *Kernel* yang lainnya dengan berbagai parameter-parameternya.

e. Proses pelatihan yang dilakukan *multi-calss* SVM menggunakan strategi 1V1 tidak sebanyak MLP, karena pada *multi-class* SVM menggunakan strategi 1V1 hanya sejumlah data latih terpilih saja yang berkonstribusi (support vector) yang akan dipelajari. Dengan demikian *multi-class* SVM menggunakan strategi 1V1 dianggap bisa lebih cepat daripada MLP.

f. Secara teoritis, *error generalisasi* yang diciptakan SVM lebih kecil daripada MLP. Solusi yang diberikan oleh SVM adalah *global optimal* sehingga selalu bisa mencapai solusi/model yang sama untuk setiap *running*. Berbeda dengan solusi/model yang diberikan oleh MLP yang *local optimal*, sehingga tidak heran bila MLP dijalankan, solusinya/modelnya selalu berbeda setiap *running*. Hal ini karena strategi pada SVM yang disebut *Structural Risk Minimization* (SRM). Dengan demikian, akurasi sebesar 97.59% yang dihasilkan oleh MLP untuk klasifikasi *wall-following robot navigation data* masih dianggap rendah, namun ternyata penelitian ini mengindikasikan hasil yang berbeda. Salah satu penyebabnya, mungkin karena SVM memang handal pada masalah klasifikasi biner namun masih dalam pengembangan yang relatif muda untuk masalah *multi-class*.

g. SVM tidak membutuhkan pemilihan parameter-parameter yang banyak. Dalam SVM, kita hanya perlu menentukan fungsi *Kernel* yang harus digunakan (untuk kasus data yang distirbusi kelasnya tidak dapat dipisahkan secara linear). Sedangkan untuk *dataset* yang berjumlah besar, SVM memang membutuhkan memori yang sangat besar untuk alokasi matriks *Kernel* yang digunakan. Namun penggunaan matriks *Kernel* mempunyai keuntungan lain, yaitu pada *dataset* dengan dimensi besar tetapi jumlah datanya sedikit akan lebih cepat karena ukuran data pada dimensi baru berkurang banyak. Hal ini merupakan alasan mengapa SVM merupakan salah satu metode yang tepat dipakai untuk memecahkan masalah berdimensi tinggi (seperti pada dataset yang digunakan dalam penelitian ini). Namun *dataset* yang digunakan dalam penelitian ini memang berdimensi tinggi namun dengan jumlah data yang besar pula, di mana secara teoritis *multi-class* SVM cukup sulit melakukan klasifikasi pada masalah berskala besar. Skala besar yang dimaksud dalam hal ini adalah jumlah sampel data yang diolah. Hal ini disebabkan karena ukuran QP yang dihasilkan. Dengan demikian, diperlukan adanya reduksi pada jumlah data di tahap *pre-processing* apabila jumlah data terlalu besar. Menurut Eko Prasetyo (2012), salah satu teknik yang bisa digunakan untuk hal tersebut adalah penyampelan. Hal ini juga sangat berguna untuk mempercapat waktu proses.

h. Ada dua pilihan untuk mengimplementasikan *multi-class* SVM, yaitu dengan menggabungkan beberapa SVM biner atau menggabungkan semua data yang terdiri dari beberapa *class* ke dalam bentuk optimasi. Dalam prakteknya, masalah *multi-class* (k>2) umumnya didekomposisi menjadi serangkaian masalah biner sehingga standar SVM dapat langsung diterapkan. Berdasarkan teori tersebut, maka penelitian ini memilih memilih cara pertama dengan strategi 1V1 yang menggunakan metode dekomposisi untuk memecahkan serangkaian masalah biner di mana sejauh ini 1V1 merupakan strategi yang terbaik daripada strategi dekomposisi lainnya karena 1V1 lebih simetris dan ukuran QP di setiap *classifier* lebih kecil. Namun dari sisi akurasi, melalui penelitian ini ternyata belum mampu memberikan akurasi yang lebih besar daripada MLP. Hal ini mungkin disebabkan karena ukuran pengklasifikasi yang diciptakan oleh pendekatan 1V1 cukup besar.

Dengan demikian, pilihan kedua dalam mengimplementasikan *multi-class* SVM dapat menjadi alternatif pada penelitian kedepannya, ataupun dengan mengguanakan strategi lainnya seperti W&W yang mengkombinasikan masalah *multiple binary-class optimization* menjadi satu fungsi tujuan tunggal dan secara bersamaan mencapai *multi-class problem* walaupun berakibat kompleksitas komputasi yang besar karena ukuran pada masalah QP, atau strategi C&S yang menyatakan bahwa eksperimen mereka mengindikasikan *state-of-the-art accuracy* pada masalah *multi-class* SVM dengan menerapkan pendekatan berbasis *Kernel* walaupun mengakibatkan kompleksitas komputasi yang besar pula. Sehingga untuk meningkatkan kinerja *multi-class* SVM, maka pengembangan lebih lanjut terhadap strategi yang bisa diterapkan pada SVM untuk mengatasi masalah *multi-class* sangat dibutuhkan.

i. Ada dua kemungkinan untuk mengolah data *non-linear* menggunakan SVM: (1) Mentransformasikan data agar *linearly separable*, baru kemudian diolah dengan SVM linear; (2) Mengolah data *non-linear* dengan SVM *non-linear*. Upaya yang kedua yang diterapkan pada penelitian ini karena proses transformasi agar data itu menjadi *linearly separable* tidak mudah dan belum tentu lebih baik kinerjanya, bisa beresiko hilangnya *critical information* yang penting untuk mengklasifikasikan data, dan karena SVM *non-linear* menghindari perhitungan transformasi ke dimensi yg lebih tinggi ini dengan cara menghitung *dot product* secara implisit melalui fungsi *Kernel*. Namun, penelitian kedepannya boleh mencoba pilihan yang pertama.

5. KESIMPULAN DAN SARAN

5.1. Kesimpulan

Berdasarkan hasil penelitian dan pembahasan, maka kesimpulan penelitian ini adalah:

a. Model *multi-class* SVM menggunakan strategi 1V1 untuk klasifikasi *wall-following robot navigation data* kurang mampu meningkatkan kecerdasan robot pada teknik *wall-following robot navigation* karena akurasinya lebih rendah (91.10%) dari pada penelitian sebelumnya menggunakan MLP (97.59%), namun dengan tingkat akurasi sebesar itu, maka keduanya masuk dalam *fair classification* sehingga keduanya dapat mempelajari navigasi robot pengikut dinding tanpa tabrakan (menjaga jarak terhadap dinding dengan baik). Sedangkan bila dipandang dari segi waktu prosesnya, secara teoritis *multi-class* SVM menggunakan strategi 1V1 lebih cepat karena hanya mempelajari data yang termasuk *support vector* saja, di mana faktor kecepatan merupakan hal yang cukup penting pada masalah navigasi robot pengikut dinding.

b. Diperolehnya model *multi-class* SVM menggunakan strategi 1V1 untuk klasifikasi *wall-following robot navigation data* dengan solusi yang *global optimal* dan tanpa perlu adanya *dimensionality reduction* dalam tingkat akurasi *fair classification*, yaitu 91.10% < 97.59% yang dihasilkan MLP karena dapat disebabkan oleh beberapa hal berikut: (1) Ketidaktepatan pemilihan fungsi *Kernel* yang berkaitan dengan berbagai nilai parameternya untuk digunakan pada SVM *non-linear*; (2) Berbeda dengan MLP yang memperlejari semua datanya, SVM hanya mempelajari data yang termasuk dalam *support vector* saja, dengan demikian dapat membuat waktu prosesnya lebih cepat namun dapat pula membuat akurasinya menurun, dimana bila semakin banyak data uji maka akurasinya semakin menurun karena tidak semua data yang dilibatkan dalam pengujiannya. (3) Kelemahan SVM dalam melakukan klasifikasi pada data yang berskala besar karena ukuran QP yang dihasilkan. Sedangkan waktu proses yang dihasilkan = 10.7505 detik. Dengan demikian, dapat diketahui bahwa dengan kinerja sebesar itu, maka model tersebut dapat mempelajari navigasi robot pengikut dinding tanpa tabrakan (menjaga jarak terhadap dinding dengan baik). Hal ini tentunya memberikan masukan bagi perkembangan ilmu pengetahuan dan teknologi, khusunya pada bidang *pattern recognition* dan robotika.

5.2. Saran

Berdasarkan hasil penelitian dan pembahasan serta kesimpulan, maka beberapa saran yang dapat diberikan adalah:

a. Pemilihan fungsi *Kernel* yang lebih tepat, karena fungsi Kernel ini akan menentukan feature space di mana fungsi classifier (hyperplane) akan dicari. Salah satu cara yang bisa diterapkan adalah dengan menggunakan metode cross validation.
b. Mentransformasikan data dalam masalah ini agar linearly separable lebih dahulu baru kemudian diolah dengan SVM linear saja.
c. Komposisi data uji sangat mempengaruhi hasil akurasi dan waktu proses dari model multi class SVM menggunakan strategi 1V1 untuk klasifikasi wall-following robot navigation data di mana bila semakin banyak data uji maka semakin buruk akurasinya namun semakin cepat waktu prosesnya. Dengan demikian, komposisi data uji yang tepat sangat dibutuhkan untuk meningkatkan kinerja dari model ini.
d. Untuk meningkatkan kinerja multi-class SVM, maka pengembangan lebih lanjut terhadap strategi yang bisa diterapkan pada SVM untuk mengatasi masalah multi-class dibutuhkan.
e. SVM cukup sulit melakukan klasifikasi pada data berskala besar karena ukuran QP yang dihasilkan. Dengan demikian, diperlukan adanya reduksi pada jumlah data di tahap pre-processing apabila jumlah data terlalu besar, misalnya dengan menggunakan teknik penyampelan. Hal ini juga sangat berguna untuk mempercapat waktu proses.
f. Menggabungkan semua data yang terdiri dari beberapa *class* ke dalam bentuk optimasi.

ACKNOWLEDGEMENT

Alhamdulillah, penulis panjatkan kehadirat Allah SWT atas berkat rahmat dan hidayah-Nya, sehingga Tesis dengan judul: “MODEL MULTI CLASS SVM MENGGUNAKAN STRATEGI 1V1 UNTUK KLASIFIKASI WALL-FOLLOWING ROBOT NAVIGATION DATA” ini dapat penulis selesaikan sesuai rencana. Penulis menyadari sepenuhnya bahwa Tesis ini tidak mungkin terwujud tanpa bantuan dan dorongan dari berbagai pihak yang tidak ternilai besarnya. Oleh karena itu, dengan segala keikhlasan dan kerendahan hati, penulis mengucapkan terima kasih dan penghargaan yang setinggi-tingginya kepada manajemen dan pihak- pihak yang tidak dapat disebutkan satu persatu dari Universitas Dian Nuswantoro dan YPIPT Ichsan Gorontalo.

PERNYATAAN ORIGINALITAS

“Saya menyatakan dan bertanggung jawab dengan sebenarnya bahwa Artikel ini adalah hasil karya saya sendiri kecuali cuplikan dan ringkasan yang masing-masing telah saya jelaskan sumbernya”.
[Azminuddin I. S. Azis – P31-2011-01146]

DAFTAR PUSTAKA

[1] T. Sutejo, Edy Mulyanto, dan Vincent Suhartono, *Kecerdasan Buatan*, Andi Offset dan Udinus Semarang, 2011.
[2] H. Anton, "Perancangan Sistem Navigasi Robot Mobil Menggunakan Kamera," *Master Theses of Electrical Engineering*, ITS, 2009.
[3] Karambakhsh A., M. Yousefi Azar Khanain, M. R. Meybodi, and A. Fakharian, "Robot Navigation Algorithm to Wall Following Using Fuzzy Kalman Filter," *IEEE International Conference on Control and Automation (ICCA)*, 2011, pp. 440-443.
[4] Fuad M., "Navigasi Berbasis Kamera RGB-D untuk Robot Pembersih Lantai dalam Koridor Pusat Robotika ITS," *Paper and Presentation of Electrical Engineering*, ITS, 2012.

[5] Nugroho A. S., W. Arief Budi Witarto, dan Dwi Handoko, "Support Vector Machine: Teori dan Aplikasinya dalam Bioinformatika," *Ilmu Komputer.Com*, 2003.

[6] F. Ananda L., Guilherme A Baretto, Marcus Veloso, and Antonio T Varela, "Short-Term Memory Mechanisms in Neural Network Learning of Robot Navigation Tasks: A Case Study," *Perception*, 2009.

[7] Wikipedia, 2012, http://id.wikipedia.org/wiki/Pengenalan_pola.

[8] H. Byun and Lee S. W., "A Survey on Pattern Recognition Applications of Support Vector Machines," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 17, 2003, pp. 459-486.

[9] H. Xisheng, Zhe Wang, Chen Jin, Yingbin Zhen, and Xiangyang Xue, "A Simplified Multi-Class Support Vector Machine with Reduced Dual Optimization," *Elsevier on Pattern Recognition Letters*, vol. 33, 2012, pp. 71-82.

[10] K. Tsuda, "Overview of Support Vector Machine," *Journal of IEICE*, vol. 83, 2000, pp. 460-466.

[11] Prasetyo Eko, *Data Mining – Konsep dan Aplikasi Menggunakan Matlab*, Andi Offset, 2012.

[12] Hsu Chih-Wei and Chih-Jen Lin, "A Comparison of Methods for Multiclass Support Vector Machines," *IEEE Trans. Neural Netw*, 2002, pp. 415-425.

[13] L. Pawan and Cory Butz, "Rough Set Based 1-v-1 and 1-v-r Approaches to Support Vector Machine Multi-Classification," *Elsevier International Journal on Information Science*, vol. 177, 2007, pp. 3782-3798.

[14] Crammer K and Yoram Singer, "On The Algorithmic Implementation of Multi-Class Kernel-Based Vector Machines," *Journal of Machine Learning Research*, vol. 2, 2001, pp. 265-292.

[15] Hsu Chih-Wei and Chih-Jen Lin, "A Simple Decomposition Method for Support Vector Machines," *Kluwer Academic Publishers on Machine Learning*, 2002, pp. 291-314.

[16] Hsu Chih-Wei and Chih-Jen Lin, "BSVM," *mloss*, 2008, http://mloss.org/software/view/62/.

[17] W. Romi S., *Data Mining: Evaluasi dan Validasi*, 2012.

[18] S. Deddy and Rony A Nugroho, "Wall Following Algorithm," *Fakultas Teknik Program Studi Teknik Elektro dan Sistem Komputer*, Universitas Kristen Satya Wacana Salatiga, 2006, pp. 1-9.

[19] Nugroho Anto S., "Diskusi Support Vector Machine," 2007, http://asnugroho.wordpress.com/2007/02/18/diskusi-mengenai-support-vector-machine/.

[20] M. Mucientes, Alcal'a R., Alcal´a J.A., and Casillas, "Learning Weighted Linguistic Rules to Control an Autonomous Robot," *International Journal of Intelligent Systems*, vol. 24, 2009, p. 226–251.

[21] Brigida, "Pengenalan Pola," 2012, http://informatika.web.id/pengenalan-pola.htm.

[22] E. Antonelo, Schrauwen B., and Stroobandt D., "Mobile Robot Control in The Road Sign Problem Using Reservoir Computing Networks," *IEEE International Conference on Robotics and Automation (ICRA'2008)*, 2008, pp. 911-916.

[23] Platt J. C., N. Cristianini, and J. S. Tylor, "Large Margin DAGs for Multi-Class Classification," *MIT Press*, 2000.

[24] Wikipedia, 2012, http://id.wikipedia.org/wiki/Klasifikasi.

[25] Wikipedia, 2013, http://en.wikipedia.org/wiki/Binary_classification.

[26] Wikipedia, "Support Vector Machine," 2013, https://en.wikipedia.org/wiki/Support_vector_machine.

[27] S. Budi, "Tutorial Support Vector Machine," *Teknik Industri*, ITS, 1995, pp. 1-23.

[28] O. Mangasarian, Musicant D., and Learn J., "Lagrangian Support Vector Machines," *Machine Learning*, 2001, p. 161–177.

[29] Yunita A., C. Fatichah, dan U. L. Yuhana, "Implementasi Metode Multiple Kernel Support Vector Machine Untuk Seleksi Fitur Dari Data Ekspresi Gen Dengan Studi Kasus Leukimia Dan Tumor Usus Besar," *Teknik Informatika*, Institut Teknologi Sepuluh Nopember Surabaya, 2010