Smart Comp: Jurnalnya Orang Pintar Komputer
Volume 14, Nomor 1, Januari 2025, hlm. 88-98
Terakreditasi Sinta, Peringkat 4, SK No. 105/E/KPT/2022
DOI : 10.30591/smartcomp.v14i1.7374

P-ISSN : 2089-676X
E-ISSN : 2549-0796

Pengenalan Permukaan Berbasis Sensor IMU Menggunakan Random Forest Classifier

Andrew Sebastian Lehman[1], Benny Budiawan Tjandrasa[2], Joseph Sanjaya[*3]
[1)]Program Studi Sistem Komputer, Fakultas Teknik, Universitas Kristen Maranatha
[2)]Program Studi Manajemen, Fakultas Bisnis, Universitas Kristen Maranatha
[3)]Program Studi Magister Ilmu Komputer, Fakultas IT, Universitas Kristen Maranatha
Email: [1]andrew.sl@eng.maranatha.edu, [2]benny.budiawan@eco.maranatha.edu,
[*3]mi1879011@student.it.maranatha.edu

(Naskah masuk: *4 Agustus 2024*, diterima untuk diterbitkan: *10 Januari 2025*)

Abstrak: *Dengan meningkatnya tantangan yang dihadapi oleh robot akibat kemajuan teknologi dan keragaman kasus, input yang diperlukan oleh robot menjadi lebih banyak dan variatif, sehingga tidak dapat diselesaikan hanya dengan if conditional yang sederhana. Oleh karena itu, penelitian ini bertujuan untuk mengembangkan model machine learning yang dapat membantu robot navigasi untuk mengidentifikasi permukaan yang optimal dengan tingkat akurasi yang tinggi. Penelitian ini menggunakan random forest sebagai algoritma machine learning yang termasuk dalam kategori supervised learning untuk menghasilkan prediksi berdasarkan data latih. Data latih di augmentasi dengan data dari yang telah disintesis menggunakan CTGAN, yaitu sebuah metode yang menggunakan jaringan saraf tiruan untuk menghasilkan data tabular heterogen yang mirip dengan data asli. Metode ini juga dapat melindungi privasi data asli dengan menggunakan privasi diferensial dan pembelajaran federasi. Setelah model machine learning dibangun, model diuji dengan data uji yang dibuat. Model machine learning yang berbasis random forest ini menunjukkan kinerja yang baik dengan mencapai akurasi 90% pada prediksi data uji.*

Kata Kunci – *Pembelajaran Mesin; Random Forest; Python; CTGAN*

IMU-Based Sensor Surface Recognition Using Random Forest Classifier

Abstract: *With the increasing challenges faced by robots due to technological advancements and diverse cases, the input required by robots has become more extensive and varied, making it impossible to rely solely on simple if-conditionals. Therefore, this study aims to develop a machine learning model to assist navigation robots in identifying optimal surfaces with high accuracy. The study employs random forest as the machine learning algorithm, categorized under supervised learning, to generate predictions based on training data. The training data is augmented with synthesized data created using CTGAN, a method that utilizes artificial neural networks to generate heterogeneous tabular data similar to the original data. This method also ensures the protection of original data privacy through differential privacy and federated learning. After building the machine learning model, it was tested using created test data. The random forest-based machine learning model demonstrated good performance, achieving 90% accuracy in test data predictions.*

Keywords – *Machine Learning; Random Forest; Python; CTGAN*

1. PENDAHULUAN

Saat ini perkembangan teknologi berjalan dengan pesat dan banyaknya kasus yang harus dihadapi menggunakan robot semakin luas serta semakin kompleks. Dengan banyaknya kasus tersebut, input yang harus diterima robot semakin banyak dan beragam sehingga kasus tersebut tidak mudah diselesaikan hanya dengan menggunakan if conditional yang telah dibuat.

Robot itu cerdas berdasarkan desain yang dibuat, untuk memahami dan menavigasikan tugas dengan benar robot tersebut membutuhkan input tentang lingkungan tersebut. Pemodelan machine

learning ini bertujuan untuk membantu robot untuk mengenali permukaan mereka berjalan dengan menggunakan data yang dikumpulkan dari Inertial Measurement Units (IMU Sensor).

Pengenalan permukaan berbasis sensor IMU menggunakan Random Forest Classifier merupakan topik yang menarik dan relevan dalam bidang teknologi sensor dan machine learning. Penelitian ini merujuk pada hasil-hasil dalam penelitian sebelumnya yang telah menunjukkan keberhasilan penggunaan Random Forest dalam berbagai aplikasi.

Wu dan rekan penelitinya mengembangkan metode navigasi robot seluler yang mengikuti dinding menggunakan Random Forest dan algoritma genetika, yang menunjukkan efisiensi dalam navigasi robot [1]. Selain itu, Kato dan rekan penelitinya menggunakan Random Forest untuk kalibrasi kesalahan pemosisian robot industri, yang membantu dalam mengidentifikasi dan mengurangi faktor kesalahan pemosisian [2]. Zhang dan rekan penelitinya mengusulkan pendekatan deteksi genggaman robot yang cepat dan akurat menggunakan pengolahan citra dan Random Forest [3]. Abdulkadium mengembangkan metode penghindaran rintangan robot menggunakan algoritma Random Forest yang disempurnakan dengan algoritma RFHTMC [4].

Gu dan Li mengusulkan Half-Voting Random Forest Algorithm untuk navigasi pejalan kaki dalam ruangan, yang meningkatkan efisiensi proses voting pada decision tree [5]. Tahir dan rekan penelitinya mengembangkan model analisis aktivitas menggunakan sensor wearable berbasis Random Forest dan SMO, yang menunjukkan efisiensi dalam analisis data olahraga dan rumah pintar [6]. Ko dan rekan penelitinya mengusulkan metode lokalisasi dalam ruangan yang kuat menggunakan filter Random Forest untuk melawan serangan spoofing MAC, yang meningkatkan akurasi prediksi dalam skenario serangan [7].

Penelitian ini bertujuan untuk mengembangkan metode pengenalan permukaan berbasis sensor IMU menggunakan Random Forest Classifier, dengan merujuk pada keberhasilan penelitian-penelitian sebelumnya dalam berbagai aplikasi Random Forest. Berdasarkan pernyataan tersebut, pada penelitian ini akan dibuat satu model machine learning menggunakan metode Random Forest. Dataset yang digunakan berdasarkan CareerCon 2019 mengenai navigasi robot. Tugas dari machine learning tersebut adalah untuk memprediksi tipe lantai apa yang menjadi permukaan lantai robot itu melaju.

2. METODE PENELITIAN

2.1. Metode Penelitian

Perancangan Sistem Dynamic Environment AR akan menggunakan model proses *waterfall* untuk mensimulasikan penempatan mebel. Dalam proses *waterfall* tersebut akan dianjurkan pendekatan pada masalah *software* yang sistematik dan sekuensial, mulai dari tingkat dan kemajuan sistem pada keseluruhan analisis, desain, kode, pengujian dan pemeliharaan [8].

Rekayasa dan pemodelan sistem informasi: perumusan hal-hal yang akan diimplementasikan ke dam sistem (*hardware*, *software*, dan lainnya)

a) Analisis: pengumpulan sumber data-data pendukung penelitian, penyusunan waktu penelitian, dan penguarian tool yang digunakan. Analisis data mencakup beberapa cabang berbeda dari statistik dan analisis yang lebih luas yang membantu menggabungkan beragam sumber data dan menemukan koneksi sambil menyederhanakan hasilnya [9].
b) Desain: perancangan representasi dari aplikasi.
c) *Coding*: penterjemahan desain ke dalam bahasa yang dapat dibaca.
d) Testing: pengujian setiap modul yang terdapat di dalam sistem, perbaikan modul apabila terjadi ketidaksesuaian dengan hasil yang diharapkan.

3. HASIL DAN PEMBAHASAN

3.1. *Navigation* Robot

Dalam konteks *mobile devices,* kemampuan bernavigasi di lingkungannya merupakan salah satu poin penting. *Mobile devices* harus dapat menghindari situasi berbahaya seperti tabrakan dan kondisi yang tidak aman (suhu, radiasi, paparan cuaca, dll.) [10].

Robot *navigation* berarti kemampuan robot untuk menentukan posisinya sendiri dalam kerangka acuannya dan kemudian merencanakan jalur menuju beberapa lokasi tujuan.

Beberapa sistem navigasi robot menggunakan pelokalan dan pemetaan secara simultan untuk menghasilkan rekonstruksi 3D di sekitarnya. Lokalisasi robot menunjukan kemampuan robot untuk menetapkan posisi dan orientasinya sendiri dalam kerangka referensi. Perencanaan jalur secara efektif merupakan perpanjangan dari pelokalan, dalam hal ini memerlukan penentuan posisi robot saat ini dan posisi lokasi tujuan, baik dalam kerangka referensi atau koordinat yang sama. Pembuatan peta bisa dalam bentuk peta metrik atau notasi apapun yang menggambarkan lokasi dalam kerangka referensi robot [11].

3.2. Percobaan Training

Pada bagian ini akan dijelaskan hasil pengamatan terhadap model *Machine Learning* pada masa training. *Machine Learning* adalah alat untuk mengubah informasi menjadi pengetahuan. Pada level yang sangat tinggi, *machine learning* adalah proses mengajar sistem komputer bagaimana membuat prediksi yang akurat ketika memasukkan data. *Machine Learning* menggunakan data dan jawaban untuk menemukan aturan di balik masalah [12].

Ada beberapa bentuk *machine learning*; *supervised, unsupervised, semi-supervised* dan *reinforcement learning*. Setiap bentuk *machine learning* memiliki pendekatan yang berbeda, tetapi semua mengikuti proses dan teori dasar yang sama.

Pada library scikit-learn phyton model *Random Forest* memiliki beberapa parameter yang dapat dilakukan tuning untuk mendapatkan performa model yang terbaik. Parameter merupakan variabel konfigurasi yang terdapat pada internal model karena variabel ini dapat diperkirakan dari data pelatihan. Algoritma memiliki mekanisme untuk mengoptimalkan parameter.

3.2.1. Based Model Random Forest

Tahap awal akan dibuat model *Random Forest* dengan parameter-parameter pada Tabel 1. Model tersebut akan digunakan pada tahap selanjutnya untuk menentukan parameter model terbaik dengan memafaatkan fasilitas *Randomized Search CV* (cross validation) pada library scikit-learn.

Tabel 1. Random Forest Parameter Tuning

Parameter	Value
n_estimator	900
max_depth	none
min_sample_split	2

Parameter *n_estimator* berguna sebagai jumlah "pohon" yang akan dibentuk terlebih dahulu sebelum dilakukan *voting* atau rata-rata prediksi, jumlah pohon yang lebih tinggi biasanya dapat menghasilkan kinerja yang lebih baik tetapi mengakibatkan proses *training* menjadi lebih lambat. Parameter *max_depth* mewakili kedalaman setiap pohon pada *Random Forest*. Semakin dalam pohon, semakin banyak perpecahan yang dimilikinya dan menangkap lebih banyak informasi tentang data. Sebuah pohon memiliki banyak analogi dalam kehidupan nyata, dan ternyata telah memengaruhi banyak pembelajaran mesin, mencakup klasifikasi dan regresi.

Parameter *min_sample_split* berguna untuk mendeskripsikan minimal jumlah titik data yang ditempatkan disebuah *node* sebelum *node* terpecah. Hasil akhir parameter dalam pembuatan base

model dapat dimunculkan dengan menggunakan fungsi "*model.get_params()*". Dengan memanfaatkan fungsi tersebut maka detail keseluruhan parameter model dapat divisualisasikan seperti pada Gambar 1.

```
In [47]: from pprint import pprint
         pprint(model.get_params())

         {'bootstrap': True,
          'ccp_alpha': 0.0,
          'class_weight': None,
          'criterion': 'gini',
          'max_depth': None,
          'max_features': 'auto',
          'max_leaf_nodes': None,
          'max_samples': None,
          'min_impurity_decrease': 0.0,
          'min_impurity_split': None,
          'min_samples_leaf': 1,
          'min_samples_split': 2,
          'min_weight_fraction_leaf': 0.0,
          'n_estimators': 900,
          'n_jobs': -1,
          'oob_score': False,
          'random_state': None,
          'verbose': 0,
          'warm_start': False}
```

Gambar 1. Detail Parameter *Base* Model

Dataset, merupakan kumpulan data. Kumpulan data mencantumkan nilai untuk setiap variabel, seperti tinggi dan berat objek, untuk setiap anggota kumpulan data. Setiap nilai dikenal sebagai datum. *Dataset* juga dapat terdiri dari kumpulan dokumen atau *file* [13].

Selanjutnya data keseluruhan perlu dibagi menjadi 2 jenis dataset yang berbeda yaitu train dan test. Pembagian dataset dilakukan dengan metode stratified random split untuk menghasilkan 2 dataset yang memiliki jumlah kelas yang sama untuk menghindari pembiasan data pada saat pembagian kedua dataset. Pembiasan ini terjadi Ketika 2 dataset hasil split memiliki jumlah yang berbeda. Pada dataset test hanya memiliki data kelas wood dan concrete. Hal ini dapat menjadi masalah karena pada saat evaluasi model tidak pernah di-training menggunakan kelas wood ataupun concrete. Setelah data terbagi, selanjutnya melanjutkan ke proses training. Hasil dari training data model ditampilkan pada Tabel 2.

Tabel 2. Hasil Training Base Model

Pengulangan	**Accuracy**
0	0.937007874015748
1	0.9133858267716536
2	0.9028871391076115
3	0.9291338582677166
4	0.905511811023622
5	0.9133858267716536
6	0.8923884514435696
7	0.910761154855643
8	0.916010498687664
9	0.89501312335958
Avg. Accuracy	**0.9115485564304464**

3.2.2. Parameter Tuning Randomized Search CV

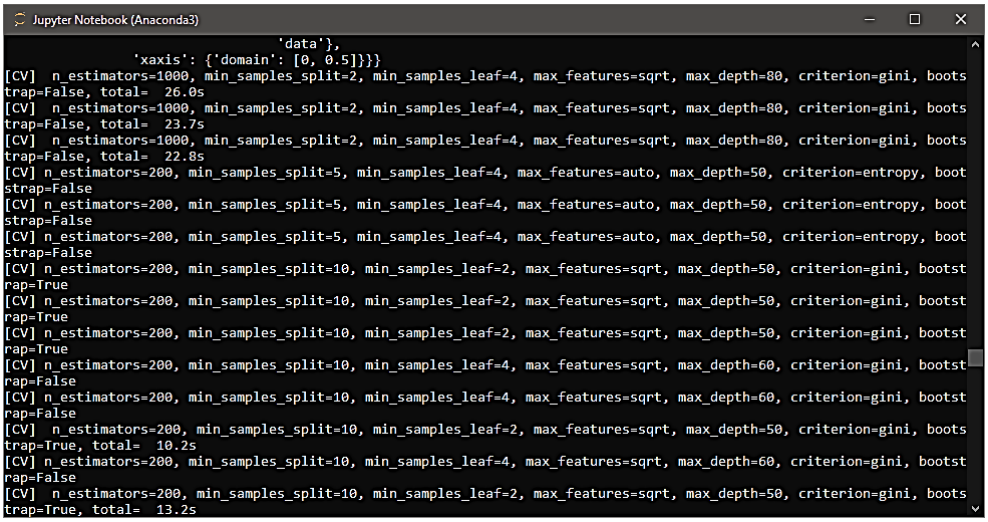
Randomized Search CV adalah algoritma yang melatih dan mengevaluasi serangkaian model dengan mengambil *random set* distribusi parameter yang telah ditentukan. Setelah itu, algoritma memilih versi paling sukses dari model yang dilihatnya setelah melatih banyak jenis versi berbeda dari model dengan kombinasi parameter yang dipilih secara acak.

Hal pertama yang perlu dilakukan adalah mendefinisikan *range* dari setiap parameter yang akan dipilih secara *random*. Parameter yang akan dipakai dalam *range* tersebut ditampilkan pada Tabel 3

Tabel 3. Hasil Training Base Model

Parameter	***Value***	**Keterangan**
criterion	['gini', *'entropy'*]	Fungsi untuk mengukur kualitas *split* . Kriteria yang didukung adalah "gini" untuk ketidakmurnian Gini dan "entropi" untuk perolehan informasi
n_estimators	[200, 400, 600, 800, 1000, 1200, 1400, 1600, 1800, 2000]	Jumlah pohon pada model.
max_features	['*auto*' , '*sqrt*']	Jumlah fitur yang perlu dipertimbangkan saat mencari pemisahan terbaik:
max_depth	[10, 20, 30, 40, 50, 60, 70, 80, 90, 100, 110, *None*]	Kedalaman *maximum* pohon pada model.
min_samples_split	[2, 5, 10]	Jumlah sampel minimum yang diperlukan untuk membagi *internal nodes*
min_samples_leaf	[1, 2, 4]	Jumlah minimum sampel yang diperlukan berada pada *leaf nodes* .

Setelah parameter yang akan diacak sudah didefinisikan, selanjutnya model di-*training* ulang dengan konfigurasi yang baru saat melakukan *training* dengan *Randomized Search CV* dapat terlihat pada log, setiap parameter yang diuji. Log tersebut divisualisasikan pada Gambar 2.



Gambar 2. Pengecekan *Randomized Parameter*

Jika diperhatikan pada Tabel 4, performa akurasi pada model meningkat. Dapat disimpulkan bahwa parameter pada *base model* bukan parameter yang paling optimal pada model. Selanjutnya perlu didapatkan parameter yang terbaik, untuk training lebih lanjut. Hal ini dapat diperoleh dengan menggunakan fungsi "*model_random.best_estimator_.get_params()*". Hasil pada proses ditampilkan pada Tabel 5.

Tabel 4. Hasil *Training Randomized Search CV*

Pengulangan	***Accuracy***
0	0.931758530183727
1	0.910761154855643
2	0.8976377952755905
3	0.9343832020997376

4	0.9212598425196851
5	0.905511811023622
6	0.9081364829396326
7	0.89501312335958
8	0.916010498687664
9	0.9002624671916011
Avg. Accuracy	**0.9120734908136482**

Tabel 5. Parameter *Tuning* Terbaik Berdasarkan *Randomized Search CV*

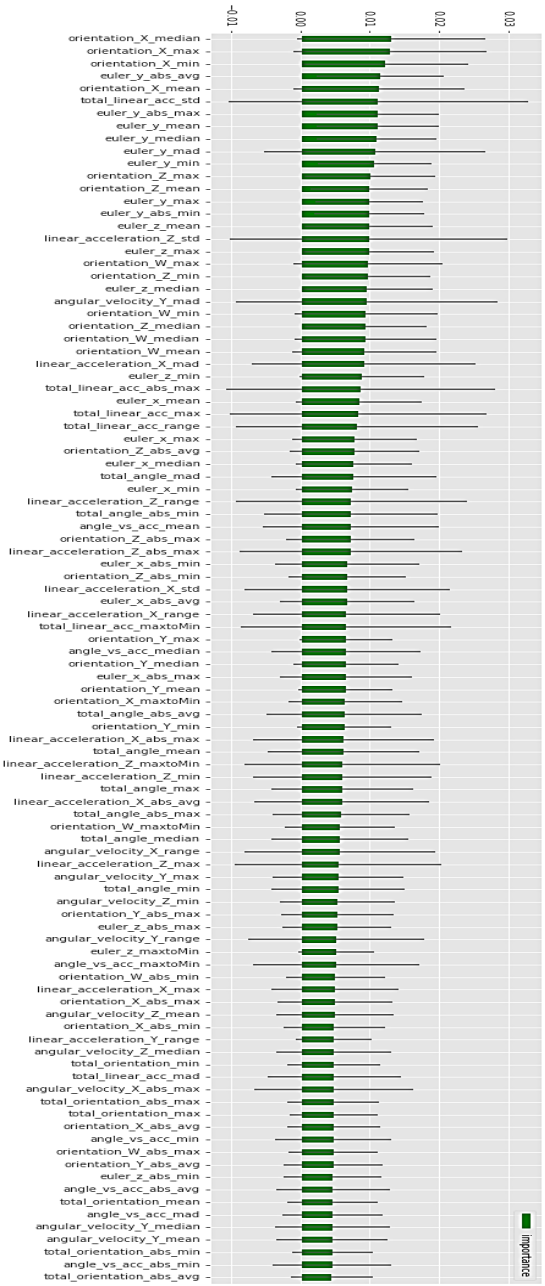
Parameter	***Value***
Criterion	gini
n_estimators	400
max_features	sqrt
max_depth	50
min_samples_split	2
min_samples_leaf	4

3.3. Pengecekan Feature Importances

Dari pengamatan terhadap tingkah *feature importancies* dari setiap fitur, hasil dapat membantu dalam penambahan akurasi model dengan membuang fitur yang tidak penting. Hal ini dikarenakan fitur yang tidak penting biasanya hanya menambahkan Tingkat kebingunan model dalam memprediksi kelas data. Pengamatan ini dapat dilakukan dengan menggunakan salah satu fitur Scikit Learn yaitu model.*feature_importances_* untuk melihat *feature importances* dari hasil training sebelumnya. Hasil dari proses pengecekan *feature importances* ditampilkan pada Tabel 6 dan divisualisasikan pada Gambar 3.

Tabel 6. *Feature Importances* Model

Features	***Importances***
orientation_X_mean	0,011178
orientation_X_median	0,012847
orientation_X_max	0,012674
orientation_X_min	0,012100
orientation_X_std	0,002291
orientation_X_range5	0,002392
orientation_X_maxtoMin	0,006277
orientation_X_mad	0,001800
orientation_X_abs_max	0,004782
orientation_X_abs_min	0,004730
orientation_X_abs_avg	0,004594
orientation_Y_mean	0,006337
orientation_Y_median	0,006378
orientation_Y_max	0,006397
orientation_Y_min	0,006145
orientation_Y_std	0,001860
orientation_Y_range	0,001938
orientation_Y_maxtoMin	0,002987
orientation_Y_mad	0,001288
orientation_Y_abs_max	0,005181



Gambar 3. *Feature Importances Model*

Selanjutnya akan pemangkasan fitur yang tidak penting. Pada konteks ini fitur yang dianggap tidak penting memiliki *feature importances* yang kurang dari 0.0025. Hal ini dapat dilakukan dengan memanfaatkan Pandas *dataframe* yaitu *data drop*. Setelah fitur yang tidak penting dibuang, akan dilakukan training ulang terhadap model *Machine Learing* dengan data yang baru. Hasil dari training ini ditampilkan pada Tabel 7.

Tabel 7. Hasil *Training* Model Setelah Pemangkasan Fitur

Pengulangan	***Accuracy***
0	0.935064935064935
1	0.9114583333333334
2	0.9373368146214099
3	0.9450261780104712
4	0.9133858267716536
5	0.9212598425196851
6	0.931758530183727

7	0.8944591029023746
8	0.8941798941798942
9	0.9202127659574468
Avg. Accuracy	**0.9204142223544931**

Jika diperhatikan pada Tabel 4 dan Tabel 7, setelah dilakukan pembuangan data fitur yang tidak berguna didapatkan penambahan performa akurasi model sebesar 0.0064935675500888.

3.4. Percobaan Implementasi Prediksi terhadap Model

Pada bagian in akan dijelaskan pengamatan terhadap hasil prediksi dari model *Machine Learning*. Data yang digunakan sebagai *input* untuk model *Machine Learning* diperoleh dengan melakukan *random* generasi data.

3.4.1. Generasi Data menggunakan Metode Conditional Generative Adversarial Network (CGAN)

Pada bagian ini akan dijelaskan proses generasi data yang akan digunakan sebagai data *input* untuk percobaan implementasi model *Random Forest*. Data tidak dapat di-*random* secara asal karena, data yang digunakan sebagai *input* harus memiliki makan agar prediksi yang dihasilkan dapt dievaluasi. Karena itu data akan digenerasi menggunakan metode *CGAN*. Data yang digenerasi mensimulasikan data baru yang tidak dikenal model [14].

Tabel 8. Hasil Generasi Data

Dataset No.	orientation_X	orientation_Y	orientation_Z	orientation_W	angular_velocity_X	angular_velocity_Y	angular_velocity_Z	linear_acceleration_X	linear_acceleration_Y	linear_acceleration_Z	Klasifikasi
1	-0,75853	-0,63435	-0,10488	-0,10597	0,10765	0,017561	0,00076741	-0,74857	2,103	-9,7532	fine_concrete
2	-0,75853	-0,63434	-0,1049	-0,106	0,067851	0,029939	0,0033855	0,33995	1,5064	-9,4128	concrete
3	-0,75853	-0,63435	-0,10492	-0,10597	0,0072747	0,028934	-0,0059783	-0,26429	1,5922	-8,7267	concrete
4	-0,75852	-0,63436	-0,10495	-0,10597	-0,013053	0,019448	-0,0089735	0,42684	1,0993	-10,096	concrete
5	-0,75852	-0,63435	-0,10495	-0,10596	0,0051349	0,0076517	0,0052452	-0,50969	1,4689	-10,441	soft_tiles
6	-0,75853	-0,63439	-0,10483	-0,1058	0,059664	0,013043	-0,013231	-0,44745	0,99281	-10,402	tiled
7	-0,75853	-0,63441	-0,10481	-0,10569	0,08214	0,044356	-0,0026956	-0,14163	0,73497	-9,4296	soft_pvc
8	-0,75852	-0,63444	-0,1048	-0,10561	0,056218	0,038162	-0,022931	-0,1216	0,075417	-8,6088	concrete
9	-0,75851	-0,63445	-0,10485	-0,10559	-0,012846	0,039004	-0,0078306	1,6	0,81611	-7,6426	hard_tiles_large_space
10	-0,75851	-0,63443	-0,10489	-0,10567	-0,090082	0,027299	-0,0099704	0,47496	0,9096	-8,812	tiled
11	-0,75848	-0,63443	-0,10497	-0,10575	-0,088418	0,011778	-0,016589	1,5943	0,37205	-11,216	soft_pvc
12	-0,75847	-0,63444	-0,10499	-0,10573	0,00067055	0,022855	-0,0057906	0,93682	0,12651	-11,273	tiled
13	-0,75846	-0,63448	-0,10495	-0,1057	0,040139	-0,0038683	-0,030181	0,67226	0,85197	-9,3933	carpet
14	-0,75846	-0,63445	-0,10504	-0,10573	-0,014728	0,0018835	-0,010793	1,4978	1,6205	-7,8959	carpet
15	-0,75842	-0,63447	-0,10512	-0,10582	-0,097677	0,0081673	-0,026073	0,091442	1,2449	-8,1673	concrete
16	-0,75838	-0,63447	-0,10525	-0,10597	-0,15602	0,01232	-0,026409	1,6563	2,1536	-9,7079	carpet
17	-0,75835	-0,63447	-0,10536	-0,10607	-0,12331	0,012997	-0,011022	0,39655	1,9488	-11,83	carpet
18	-0,75835	-0,63451	-0,10518	-0,10598	-0,008814	-0,030847	-0,021756	0,32796	1,2196	-12,512	soft_tiles
19	-0,75835	-0,63453	-0,10511	-0,10595	0,050935	-0,024422	-0,018136	0,77442	2,6077	-9,5633	tiled
20	-0,75833	-0,63456	-0,10512	-0,10596	-0,0052109	0,0051752	-0,026287	0,43819	3,6726	-6,8334	soft_tiles

CGAN adalah arsitektur untuk melatih model generative berbasis *deep learning*. Arsitekturnya terdiri dari model generator dan model diskriminator. Model generator bertanggung jawab untuk menghasilkan data baru yang masuk akal. Model diskriminator bertanggung jawab untuk mengklasifikasikan data yang diberikan sebagai nyata (diambil dari *dataset*) atau palsu (degenerasi). Pada projek ini akan digunakan *library* "ctgan". Instalasi *library* ini dapat dilakukan menggunakan *command "pip install ctgan"*. Hasil generasi data ditampilkan pada Tabel 8.

Kolom "Dataset No." akan digunakan sebagai identifikasi *set* data untuk prediksi pada bagian selanjutnya. Kolom yang lainnya selain kolom klasifikasi adalah kolom fitur data yang akan digunakan sebagai *input* model. Kolom "Klasifikasi" adalah kolom yang berguna sebagai nilai klasifikasi berdasarkan data *input*, kolom ini akan digunakan untuk membandingkan hasil prediksi model sehingga model dapat dievaluasi.

3.4.2. Percobaan Prediksi terhadap Data Generasi

Pada bagian ini akan dijelaskan hasil percobaan prediksi menggunakan model yang sudah dibuat terhadap data yang telah digenerasi pada bagian sebelumnya. Hasil dari prediksi model terhadap generasi data pada Tabel 8 ditampilkan pada Tabel 9.

Tabel 9. Hasil Prediksi Model terhadap Generasi Data

Dataset No.	Klasifikasi	Prediksi	Status
1	*fine_concrete*	*fine_concrete*	Benar
2	*concrete*	*concrete*	Benar
3	*concrete*	*concrete*	Benar
4	*concrete*	*concrete*	Benar
5	*soft_tiles*	*concrete*	Salah
6	*tiled*	*carpet*	Salah
7	*soft_pvc*	*soft_pvc*	Benar
8	*concrete*	*concrete*	Benar
9	*hard_tiles_large_space*	*hard_tiles_large_space*	Benar
10	*tiled*	*tiled*	Benar
11	*soft_pvc*	*soft_pvc*	Benar
12	*tiled*	*tiled*	Benar
13	*carpet*	*carpet*	Benar
14	*carpet*	*carpet*	Benar
15	*concrete*	*concrete*	Benar
16	carpet	carpet	Benar
17	carpet	carpet	Benar
18	soft_tiles	soft_tiles	Benar
19	tiled	tiled	Benar
20	soft_tiles	soft_tiles	Benar

Jika diperhatikan pada Tabel 9, dari 20 data *sample* yang digenerasi pada bagian sebelumnya terjadi 2 kesalahan prediksi. Sehingga disimpulkan bahwa model pada percobaan implementasi terhadap data yang tidak dikenali memiliki performa akurasi 90%.

4. KESIMPULAN

Berdasarkan hasil pembuatan dan pengujian model *machine learning* untuk membantu model robot navigasi untuk mengenali permukaan lantai yang telah dibuat, terdapat beberapa kesimpulan yang dapat diambil:

1. Pembuatan model *machine learning* telah berhasil direalisasikan.
2. Model *machine learning* yang dirancang berdasarkan *Random Forest* terhadap *data test* berhasil mencapai rata-rata akurasi 92%.
3. Model *machine learning* yang dirancang berdasarkan *Random Forest* berhasil mencapai akurasi 90% pada saat prediksi terhadap data baru yang tidak dikenali.
4. Model *Random Forest* dapat mencapai performa akurasi maksimal terhadap data yang dipakai dengan melakukan perubahan terhadap parameter *criterion* menjadi gini, *n_estimator* menjadi 400, *max_features* menjadi *sqrt*, *max_depth* menjadi 50, *min_samples_split* menjadi 2 dan *max_samples leaf* menjadi 4.

DAFTAR PUSTAKA

[1] P. Wu, M. Fang, and Z. Ding, "Wall-Following Navigation for Mobile Robot Based on Random Forest and Genetic Algorithm," in Intelligent Computing Theories and Application, vol. 12837, D.-S. Huang, K.-H. Jo, J. Li, V. Gribova, and A. Hussain Eds. Cham: Springer International Publishing, 2021, pp. 122-131.

[2] D. Kato *et al.*, "Positioning Error Calibration of Industrial Robots Based on Random Forest," (in en), *International Journal of Automation Technology*, vol. 15, no. 5, pp. 581-589, 2021-09-05 2021, doi: 10.20965/ijat.2021.p0581.

[3] J. Zhang, M. Li, Y. Feng, and C. Yang, "Robotic grasp detection based on image processing and random forest," (in en), *Multimedia Tools and Applications*, vol. 79, no. 3-4, pp. 2427-2446, 01/2020 2020, doi: 10.1007/s11042-019-08302-9.

[4] A. M. Abdulkadium, "A Robot Obstacle Avoidance Method based on Random Forest HTM Cortical Learning Algorithm," *Webology*, vol. 17, no. 2, pp. 788-803, 2020-12-21 2020, doi: 10.14704/WEB/V17I2/WEB17067.

[5] Z. Gu and Q. Li, "Half-Voting Random Forest Algorithm and Its Application in Indoor Pedestrian Navigation," (in en), *Automatic Control and Computer Sciences*, vol. 54, no. 2, pp. 100-109, 03/2020 2020, doi: 10.3103/S0146411620020054.

[6] S. Badar ud din Tahir, A. Jalal, and M. Batool, "Wearable Sensors for Activity Analysis using SMO-based Random Forest over Smart home and Sports Datasets," in *2020 3rd International Conference on Advancements in Computational Sciences (ICACS)*, 2/2020 2020, Lahore, Pakistan: IEEE, pp. 1-6, doi: 10.1109/ICACS47775.2020.9055944.

[7] D. Ko, S.-H. Choi, S. Ahn, and Y.-H. Choi, "Robust Indoor Localization Methods Using Random Forest-Based Filter against MAC Spoofing Attack," (in en), *Sensors*, vol. 20, no. 23, p. 6756, 2020-11-26 2020.

[8] C. A. Crespo-Santiago and S. d. l. C. D. Cosme, "Waterfall method: a necessary tool for implementing library projects," *HETS Online Journal*, vol. 1, no. 2, pp. 86-99, 2011 2011.

[9] H. J. Adèr, G. J. Mellenbergh, and D. J. Hand, *Advising on research methods: a consultant's companion*. Huizen: Johannes van Kessel (in eng), 2008, p. 574.

[10] C. Chen, W. Chai, A. K. Nasir, and H. Roth, "Low cost IMU based indoor mobile robot navigation with the assist of odometry and Wi-Fi using dynamic constraints," in *2012 IEEE/ION Position, Location and Navigation Symposium - PLANS 2012*, 04/2012 2012, Myrtle Beach, SC, USA: IEEE, pp. 1274-1279, doi: 10.1109/PLANS.2012.6236984.

[11] A. Chatterjee, A. Rakshit, and N. N. Singh, *Vision based autonomous robot navigation: algorithms and implementations* (Studies in computational intelligence, no. 455). Heidelberg ; New York: Springer, 2013, p. 226.

[12] P. Domingos, *The master algorithm: how the quest for the ultimate learning machine will remake our world*. New York: Basic Books, a member of the Perseus Books Group, 2015, p. 329.

[13] V. Nationen, V. Nationen and V. Nationen, Eds. *Impact on data quality* (Statistical data editing, no. United Nations Statistical Commission ... ; 3). New York, NY: United Nations (in eng), 2006, p. 366.
[14] O. Juwita, N. D. Aprilianti, K. Wibowo, and M. F. Najib, "Pengolahan Sampah Plastik Menjadi Eco Paving Block di Desa Pekauman Bondowoso," *JAST : Jurnal Aplikasi Sains dan Teknologi*, vol. 8, no. 1, pp. 73-81, 2024-07-10 2024, doi: 10.33366/jast.v8i1.5904.