HOAQ: JURNAL TEKNOLOGI INFORMASI, Volume 15 Nomor 2, Desember 2024, p107-115
p-ISSN: 2337-5280, e-ISSN: 26207427
DOI: https://doi.org/10.52972/hoaq.vol15no2.p107-115

IMPLEMENTASI PATH SMOOTHING MENGGUNAKAN ALGORITMA GRADIENT DESCENT MENGGUNAKAN SIMULASI V-REP

Akhmad Jayadi[1], Dwi Handoko[2], Rizka Permata[3], Atika Arpan[4], Dian Meilantika[5]

[12345]Program Studi Manajemen Informatika, Politeknik Negeri Lampung
Jl. Soekarno-Hatta, Bandar Lampung – Lampung, Indonesia
Email: [1]akhmad.jayadi@polinela.ac.id, [2]dwihandoko@polinela.ac.id, [3]rizka.permata@polinela.ac.id, [4]atika.arpan@polinela.ac.id, [5]dianmeilantika@polinela.ac.id

ABSTRAK

Perkembangan teknologi robot *mobile* telah mengubah berbagai sektor industri, dan salah satu tantangan utama dalam pengoperasian robot *mobile* adalah perencanaan jalur (*path planning*). Jalur yang direncanakan sering kali tidak mempertimbangkan gerakan robot, terutama saat menghadapi belokan tajam yang dapat menyebabkan inefisiensi gerak robot, untuk mengatasi hal ini, teknik *path smoothing* digunakan untuk mereduksi ketajaman belokan, memungkinkan robot bergerak lebih lancar dan efisien. Salah satu algoritma yang digunakan dalam *path smoothing* adalah *Gradient Descent*, yang secara iteratif memperbaiki jalur dengan meminimalkan jarak antara titik-titik. Simulasi menggunakan platform V-REP sangat berguna untuk menguji efektivitas algoritma, dengan harapan dapat meningkatkan kinerja navigasi robot dan berkontribusi pada pengembangan robot *mobile* yang lebih efisien dan aman. Setelah jalur optimal ditentukan, langkah selanjutnya adalah memperlancar lintasan menggunakan path smoothing. Proses ini mengoptimalkan koordinat yang dihaluskan (y) untuk mengurangi jarak antara jalur asli (x) dan jalur yang dihaluskan dengan persamaan $min = (x_i - y_i)^2$ Pembaruan pertama memindahkan y untuk meminimalkan jarak antara koordinat asli dan yang dihaluskan, sedangkan pembaruan kedua mengurangi penyimpangan antara koordinat yang berdekatan, dengan parameter beta yang mengontrol pembaruan tersebut. Hasilnya, jalur menjadi lebih mulus dan efisien untuk dilalui robot. Hasil dari penelitian yang telah dilakukan menunjukan algoritma yang diterapkan telah mampu membuat jalur yang lebih halus dari sebelum diterapkannya algoritma ini.

Kata kunci: *Mobile Robot; Path Planning; Gradient Descent*

ABSTRACT

The development of mobile robot technology has transformed various industrial sectors, with one of the main challenges in mobile robot operation being path planning. Planned paths often do not take into account the robot's movement, especially when facing sharp turns that can cause inefficiencies in the robot's motion. To address this, path smoothing techniques are used to reduce the sharpness of turns, allowing the robot to move more smoothly and efficiently. One algorithm used in path smoothing is Gradient Descent, which iteratively refines the path by minimizing the distance between points. Simulations using the V-REP platform are very useful for testing the effectiveness of the algorithm, with the aim of improving robot navigation performance and contributing to the development of more efficient and safer mobile robots. After the optimal path is determined, the next step is to smooth the trajectory using path smoothing. This process optimizes the smoothed coordinates (y) to reduce the distance between the original path (x) and the smoothed path using the equation $min = (x_i - y_i)^2$. The first update moves y to minimize the distance between the original and smoothed coordinates, while the second update reduces deviations between adjacent coordinates, with the beta parameter controlling the update. As a result, the path becomes smoother and more efficient for the robot to follow. The results of the research indicate that the applied algorithm has successfully created a smoother path compared to the original path before the algorithm was applied.

Keywords: Mobile Robot; Path Planning; Gradient Descent;

1. PENDAHULUAN

Perkembangan teknologi robotika *mobile* telah mengubah berbagai sektor industri, mulai dari logistik, transportasi, hingga eksplorasi. Salah satu tantangan utama dalam pengoperasian robot mobile adalah perencanaan jalur atau path planning. Robot harus mampu bergerak dengan efisien dari titik awal menuju titik tujuan yang telah ditentukan, dengan mempertimbangkan berbagai faktor seperti keberadaan halangan dan kelengkungan jalur. Pada

umumnya, jalur yang direncanakan oleh algoritma perencanaan jalur akan menghubungkan kedua titik tersebut, namun tidak selalu mempertimbangkan kenyamanan gerakan robot, terutama ketika harus melewati belokan tajam.

Beberapa jenis robot mobile memiliki keterbatasan dalam hal manuver, terutama saat melewati belokan tajam [1]. Ketika sebuah robot mengikuti jalur yang memiliki sudut belokan yang sangat tajam , robot tersebut cenderung mengalami pergeseran posisi yang besar untuk menyesuaikan arah heading-nya. Hal ini bisa menyebabkan inefisiensi dalam gerakan dan bahkan kecelakaan jika robot tidak dapat menyesuaikan arah dengan baik. Oleh karena itu, penghalusan jalur atau path smoothing menjadi penting untuk membuat jalur yang awalnya tajam menjadi lebih halus dan lebih mudah dilalui oleh robot.

Path smoothing adalah teknik yang digunakan untuk mereduksi ketajaman belokan pada jalur yang direncanakan sehingga robot dapat bergerak dengan lebih lancar. Dengan cara ini, robot dapat menavigasi belokan yang lebih lembut dan mengurangi kemungkinan terjadinya kecelakaan atau kegagalan saat melintasi jalur. Selain itu, penghalusan jalur juga dapat meningkatkan efisiensi perjalanan robot, karena jalur yang lebih halus sering kali mengurangi waktu perjalanan dan mengoptimalkan penggunaan energi. Salah satu tantangan dalam path smoothing adalah bagaimana membuat jalur yang lebih halus namun tetap menjaga agar jalur tersebut aman, efisien, dan tetap menghindari halangan yang ada di sekitarnya.

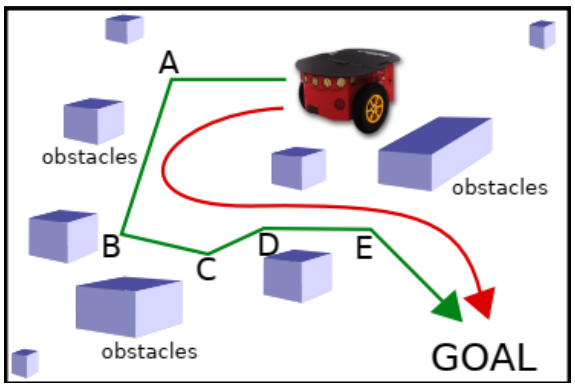
Berbagai algoritma telah dikembangkan untuk mengatasi masalah path smoothing, seperti pada penelitian [2] yang menerapkan BFS-Path Smoothing yang dapat menghubungkan node start dengan node goal tanpa melalui node tengah dengan mempertimbangkan mekanisme path smoothing yang dibangun. Pada penelitian [3] mengimplementasikan Path Smoothing menggunakan algoritma *Support Vector Machine* (SVM) untuk melengkapi data yang hilang pada moving object. Serta algoritma path smoothing yang digunakan pada penelitian ini adalah algoritma Gradient Descent. Algoritma ini merupakan metode optimasi yang digunakan untuk mencari solusi terbaik dengan menuruni gradien atau kemiringan dari fungsi yang dioptimalkan [4]. Dalam path smoothing, algoritma Gradient Descent dapat diterapkan untuk mencari jalur yang lebih optimal dengan meminimalkan jarak antara titik-titik yang dilalui oleh robot, sekaligus memastikan jalur tersebut tidak melanggar batasan-batasan seperti halangan di sekitarnya. Kelebihan utama dari algoritma ini adalah kemampuannya untuk secara iteratif melakukan perbaikan terhadap jalur yang ada, sehingga menghasilkan jalur yang lebih optimal. Dalam konteks SVM yang digunakan untuk melengkapi data yang hilang pada moving object, SVM berfokus pada klasifikasi dan prediksi data, yang lebih berorientasi pada analisis data daripada pengoptimalan jalur secara langsung. Di sisi lain, Gradient Descent lebih cocok untuk optimasi jalur, menghubungkan titik awal dan tujuan dengan jalur yang lebih efisien dan halus.

Simulasi pada platform seperti V-REP (*Virtual Robot Experimentation Platform*) [5] menjadi sangat berguna dalam penelitian ini. V-REP memungkinkan simulasi yang realistis dan interaktif dari pergerakan robot dalam lingkungan yang kompleks. Dengan menggunakan V-REP, berbagai skenario dapat diuji coba untuk melihat seberapa efektif algoritma path smoothing berbasis Gradient Descent dalam mengatasi masalah pergerakan robot. Platform ini juga memberikan kemudahan dalam visualisasi jalur yang dilalui oleh robot dan memungkinkan peneliti untuk menguji kinerja algoritma dalam situasi yang beragam, baik dalam hal konfigurasi jalur, posisi halangan, maupun karakteristik robot itu sendiri. Oleh karena itu, penelitian ini bertujuan untuk mengeksplorasi implementasi *path smoothing* menggunakan algoritma Gradient Descent dalam simulasi V-REP, dengan harapan dapat meningkatkan kinerja navigasi robot dan memberikan kontribusi dalam pengembangan teknologi robotika mobile yang lebih efisien dan aman.

2. METODE PENELITIAN

Path Smoothing

Path smoothing adalah upaya untuk mengurangi sudut lancip yang ada pada path, dengan kata lain, path smoothing digunakan untuk membentuk lengkungan pada lintasan. Path smoothing sangat berguna ketika diimplementasikan pada lingkungan yang diskrit seperti halnya grid world. Sebuah perancangan dalam membuat belokan yang melengkung akan menjadi solusi ideal dalam *vehicle*, karena beberapa jenis vehicle tidak dapat berbelok 90 derajat, seperti contohnya adalah mobil bus maupun pesawat terbang jenis *fixed-wing*. Beberapa alasan dalam *path smoothing* adalah memungkinkan izin untuk memungkinkan kesalahan yang secara alami terjadi melalui proses pergerakan, kesalahan data sensorik dalam menentukan posisi kendaraan, dan tindakan bergerak itu sendiri memiliki kemungkinan mengakumulasi beberapa kesalahan.



Gambar 1. The Smoothed Path [6]

Jalur yang dihasilkan oleh sebagian besar algoritma perencanaan jalur akan menghasilkan jalur yang terdiri dari garis lurus dan belokan tajam. Misalnya, Gambar 1 menunjukkan robot dan posisi targetnya. Garis hijau adalah garis yang tersusun atas garis lurus dan belokan tajam di titik A, B, C, D, dan E. Jalur ini tidak diinginkan untuk pergerakan robot, karena robot tidak bisa tiba-tiba berbelok tajam, sehingga perlu dilakukan perlambatan. Dalam kasus kursi roda robotik yang dioperasikan oleh orang yang cedera atau pasien, berhenti tiba-tiba dan belokan tajam seperti itu tidak menguntungkan, dan bahkan dapat menyebabkan cedera atau perasaan tidak menyenangkan. Selain itu, dari sudut pandang orang luar di lingkungan yang sama, gerakan robot yang tidak stabil tersebut merupakan hal yang tidak wajar, dan sulit untuk memprediksi posisi robot berikutnya untuk menghindari tabrakan. Dalam beberapa kasus, bergantung pada kinematika robot, bahkan mungkin sulit untuk membuat belokan tajam.

Beberapa algoritma path smoothing diantaranya adalah Hermite interpolation [7], cubic splines interpolation [8], B-spline interpolation [9], Bezier curve based dan Gradient Descent. Pada penelitian ini algoritma yang diimplementasikan adalaha Gradient Descent.

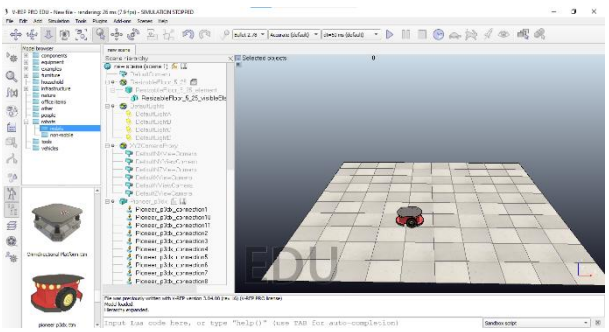
Gradien Descent

Jalur yang direncanakan oleh sebagian besar algoritma perencanaan jalur mungkin berisi belokan tajam yang sulit diikuti oleh robot non-holonomis, itulah mengapa dibutuhkan penghalusan jalur untuk mengurangi belokan tajam ini. Pada dasarnya tujuannya adalah untuk meminimalkan jarak antara titik sekarang dan titik berikutnya, dan juga meminimalkan jarak titik tersebut menjauh dari posisi semula, hal melakukannya dengan menggunakan penurunan gradien.

Simulasi VREP

V-REP adalah perangkat lunak yang digunakan untuk mensimulasikan robot dalam lingkungan 3D. Beberapa contoh desain robot yang tersedia dalam V-REP antara lain Robot Humanoid NAO, Robot Beroda Pioneer P3DX, Epuck, dan Robot Manipulator Lengan [10]. V-REP dapat dioperasikan menggunakan Remote API untuk kendali jarak jauh melalui aplikasi eksternal seperti Python, Java, atau MATLAB [11]. Sebelum memanfaatkan Remote API, skrip anak (*child scripts*) harus dinonaktifkan untuk mencegah konflik antara kendali yang dilakukan dengan Python dan kendali internal V-REP. Dalam penelitian ini, sebuah lorong dengan dinding yang diletakkan secara acak dibuat sebagai bagian dari simulasi.

.


Gambar 2. Lingkungan V-REP

Gambar 2 menyajikan antarmuka penggunaan V-REP dengan beberapa bilah elemen seperti menu bar, toolbars, dll. Banyaknya pilihan robot yang dapat digunakan seperti robot mobile dan non-mobile, komponen, peralatan, furnitur, dll yang memfasilitasi pengguna untuk dapat lebih dekat dengan dunia sesungguhnya dengan simulasi yang andal dan efektif [12].

3. HASIL DAN PEMBAHASAN

Asumsikan bahwa sudah memiliki jalur yang optimal untuk dilalui oleh robot mobile, tetapi kita ingin memperlancar lintasannya. Untuk persamaan ini, y mewakili koordinat yang dihaluskan pada langkah waktu i sedangkan x mewakili koordinat asli yang tidak mulus. Elemen ke-i dari x mengacu pada daftar yang berisi koordinat x dan y dari sebuah objek pada langkah waktu ke-i.
Optimasi pertama adalah meminimalkan jarak antara koordinat jalur asal (x) dan jalur yang dihaluskan (y).

$$min\ (x_i - y_i)^2 \quad (1)$$

Untuk pembaruan pertama, kita dapat memperbarui koordinat yang dihaluskan ke arah yang meminimalkan jarak antara koordinat yang tidak dihaluskan dan koordinat yang dihaluskan.

$$y_i = y_i + \alpha(x_i - y_i) \quad (2)$$

Untuk pembaruan kedua, kita dapat memperbarui koordinat yang dihaluskan ke arah yang meminimalkan penyimpangan antara koordinat yang dihaluskan yang berdekatan. Beta bertindak seperti alfa yang menyesuaikan seberapa besar penekanan untuk diberikan pada pembaruan koordinat yang diperhalus.

$$y_i = y_i + \beta(y_{i+1} + y_{-1} - 2y_i) \quad (3)$$

Jalur yang lebih mulus dapat dicapai dengan mengoptimalkan kondisi yang ditentukan dalam (2) dan (3).
Berikut merupakan *pseudocode* dari implementasi algoritma.

1. Input
   Langkah awal adalah melakukan inisisasi terhadap beberapa parameter masukan seperti x, y, alpha, dan beta. Seperti yang terlihat pada Gambar 3.

```
# x = jalur asli yang tidak mulus (koordinat titik-titik jalur)
# y = jalur yang dihaluskan, inisialisasi dengan x (y = x pada awalnya)
# alpha = parameter untuk pembaruan pertama (kontrol seberapa besar langkah perbaikan)
# beta = parameter untuk pembaruan kedua (kontrol seberapa besar penekanan pada koordinat
```

Gambar 3. Proses Input

2. Jalur yang diberikan
   Langkah ke-2 adalah memasukan titik jalur untuk selanjutnya nanti akan dirubah titiknya. Titik koordinatnya seperti yang terlihat pada Gambar 4.

```
# Jalur yang diberikan:
x = [439, 67]   # Contoh koordinat titik pada jalur yang tidak halus
y = x           # Inisialisasi jalur yang dihaluskan dengan jalur asli
```

Gambar 4. Proses Inisiasi Titik

3. Parameter
   Langkah ke-3 adalah memasukan parameter alpha dan beta untuk menentukan besar langkah dalam pembaruan pertama dan besar penekanan dalam pembaruan kedua. Parameter alpha yang digunakan adalah sebesar 0.1 dan beta sebesar 0.05 seperti yang terlihat pada Gambar 5.

```
# Parameter
alpha = 0.1     # Menentukan seberapa besar langkah untuk pembaruan pertama
beta = 0.05     # Menentukan seberapa besar penekanan untuk pembaruan kedua
```

Gambar 5. Proses Inisiasi Parameter

4. Iterasi untuk memperhalus jalur
   Langkah ke-4 adalah melakukan iterasi, proses ini dilakukan dalam beberapa iterasi (misalnya 100 iterasi) untuk memperhalus jalur hingga mencapai jalur yang lebih lancar. Seperti yang terlihat pada Gambar 6.

```
for i from 1 to 100:
    # Pembaruan pertama: Meminimalkan jarak
    y[i] = y[i] + alpha * (x[i] - y[i])

    # Pembaruan kedua: Meminimalkan penyimpangan
    if i > 1 and i < length(x):
        y[i] = y[i] + beta * (y[i+1] + y[i-1] - 2 * y[i])
```

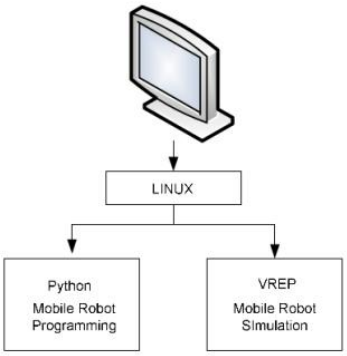
Gambar 6. Proses Iterasi

5. Output
   Langkah ke-5 yaitu hasil akhir berupa jalur yang telah dihasilkan, seperti yang terlihat pada Gambar 7. Dengan hasilnya sebesar [439, 67].

```
# Output:
# y = jalur yang telah dihaluskan
print("Jalur yang dihaluskan: ", y)
```

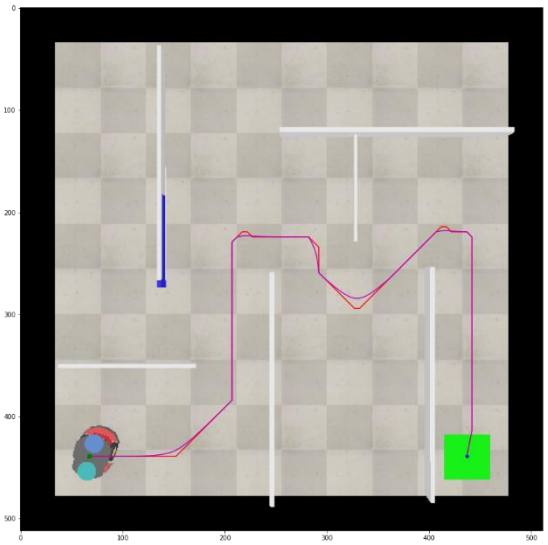
Gambar 7. Hasil Luaran

Di sini peneliti menggunakan komputer berbasis Windows, V-REP untuk mensimulasikan lingkungan mobile robot, sedangkan skema smoothing diimplementasikan dengan Python seperti yang diilustrasikan pada Gambar 8.



Gambar 8. Experimental Setup

Hasilnya, saat mobile robot mencapai tujuan yang ditentukan, dapat dilihat pada Gambar 9.



Gambar 9. *The mobile robot reaches the goal*

Gambar 9 adalah hasil dari apa yang telah dilakukan selama ini, kami menandai pusat gawang dan pusat robot, kami menggambar jalur yang direncanakan dengan warna merah dan jalur yang dihaluskan dalam warna magenta.

Dalam proses ini, jalur awal yang berbentuk poligonal atau memiliki sudut tajam dihaluskan sehingga menjadi lebih mulus, meminimalkan perubahan mendadak dalam arah lintasan. Hal ini bertujuan untuk meningkatkan efisiensi navigasi robot dan mengurangi energi yang diperlukan untuk melakukan pergerakan.

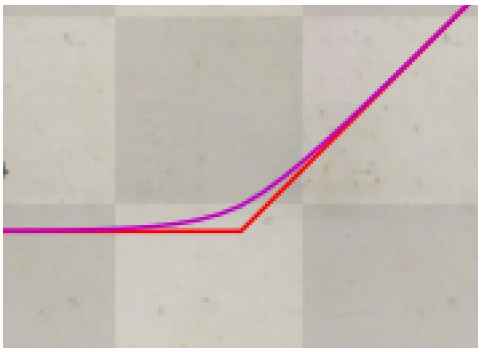
Pada tahap simulasi, dilakukan pengamatan terhadap perubahan jalur sebelum dan sesudah diterapkan algoritma. Data yang dihasilkan menunjukkan perbandingan koordinat lintasan awal (*Path*) dengan lintasan hasil perbaikan (*Newpath*). Sebagai contoh, pada belokan pertama, terlihat bahwa lintasan awal memiliki sudut yang tajam, sedangkan lintasan hasil smoothing lebih mulus dengan pengurangan deviasi sudut. Hal ini ditunjukkan pada tabel data yang merepresentasikan koordinat jalur sebelum dan sesudah *smoothing* pada belokan kedua.

Hasilnya menunjukkan bahwa algoritma Gradient Descent mampu menghasilkan jalur yang lebih halus tanpa kehilangan informasi posisi penting, sekaligus memperbaiki kelayakan lintasan untuk aplikasi nyata pada robot otonom. Implementasi ini memberikan kontribusi penting dalam pengembangan algoritma path planning untuk meningkatkan akurasi dan efisiensi gerak.

Seperti yang terlihat pada Tabel 1, 2, 3, 4 dan 5 yang merupakan tabel perbandingan antara koordinat jalur sebelum dilakukan penghalusan serta Gambar 10, 11, 12, 13 dan 14.

Tabel 1. Data perbandingan sebelum dan sesudah pada belokan pertama

Path	Newpath
[439 67]	[439. 67]
[439 72]	[438.99494948 72]
[439 77]	[438.98904537 77]
[439 82]	[438.98130282 82]
[439 87]	[438.97043131 87]
[439 92]	[438.95461929 92]
[439 97]	[438.93123212 97]
[439 102]	[438.89637285 102]
[439 107]	[438.84423265 107]
[439 112]	[438.76612259 112]
[439 117]	[438.6490254 117]
[439 122]	[438.47342592 122]
[439 127]	[438.21005857 127]
[439 132]	[437.81502966 132]
[439 137]	[437.22250186 137]
[439 142]	[436.33372124 142]
[439 147]	[435.00055831 147]
[439 152]	[433.00081978 152]
[434 157]	[430.00121632 157]
[429 162]	[426.33514773 162]
[424 167]	[422.22493621 167]
[419 172]	[417.81888009 172]
[414 177]	[413.2159702 177]
[409 182]	[408.4823884 182]
[404 187]	[403.66253781 187]
[399 192]	[398.78644344 192]
[394 197]	[393.87475628 197]
[389 202]	[388.94219518 202]



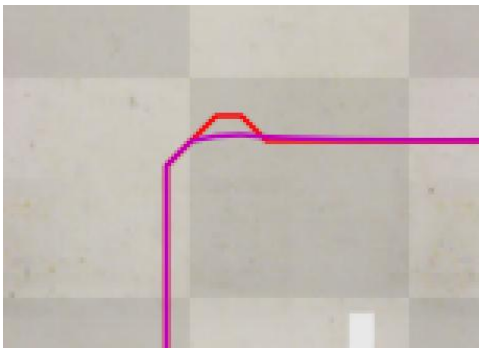
Gambar 10. Belokan Pertama

Tabel 2. Data perbandingan sebelum dan sesudah pada belokan kedua

Path	Newpath
[219 217]	[223.07409992 217]
[219 222]	[222.82721646 222]

Path	Newpath
[224 227]	[223.2182024 227]
[224 232]	[223.47888871 232]
[224 237]	[223.65272312 237]
[224 242]	[223.76867804 242]
[224 247]	[223.8460793 247]
[224 252]	[223.89782709 252]
[224 257]	[223.93254606 257]
[224 262]	[223.95602271 262]
[224 272]	[223.98367853 272]
[224 277]	[223.99246701 277]

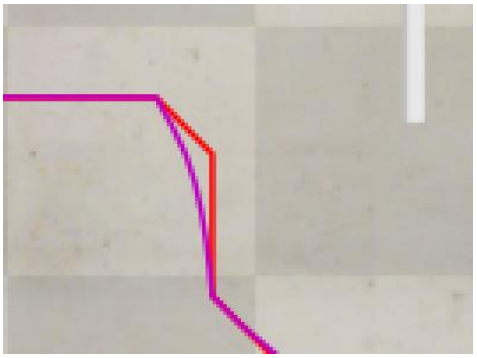
Perubahan terjadi pada sumbu x dari 219 menjadi 223.07, menunjukkan perpindahan kecil untuk memperhalus jalur. Pada koordinat [224 232], terlihat sumbu x menyesuaikan dari 224 menjadi 223.47, memperlihatkan penyelarasan agar lintasan lebih linier dan melengkung secara mulus. Pada koordinat [224 277] di sini, sumbu x hanya sedikit bergeser dari 224 ke 223.99, hampir tidak ada perubahan pada sumbu y. Hal ini menunjukkan stabilitas lintasan pada bagian akhir.



Gambar 11. Belokan Kedua

Tabel 3. Data perbandingan sebelum dan sesudah pada belokan ketiga

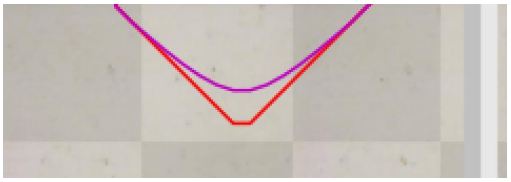
Path	Newpath
[224 282]	[224. 282]
[229 287]	[229. 284.80880861]
[234 292]	[234. 287.25241866]
[239 292]	[239. 288.90476515]
[244 292]	[244. 290.04123917]
[249 292]	[249. 290.85125305]
[254 292]	[254. 291.4698091]
[259 292]	[259. 292]



Gambar 12. Belokan Ketiga

Tabel 4. Data perbandingan sebelum dan sesudah pada belokan keempat

Path	Newpath
[264 297]	[264. 297]
[269 302]	[268.27214481 302]
[274 307]	[272.42297996 307]
[279 312]	[276.31097802 312]
[284 317]	[279.75080538 317]
[289 322]	[282.48243334 322]
[294 327]	[284.12779998 327]
[294 332]	[284.12779979 332]
[289 337]	[282.4824328 337]
[284 342]	[279.75080454 342]
[279 347]	[276.31097702 347]
[274 352]	[272.422979 352]
[269 357]	[268.27214415 357]
[264 362]	[264. 362]



Gambar 13. Belokan Keempat

Tabel 5. Data perbandingan sebelum dan sesudah pada belokan kelima

Path	Newpath
[219 407]	[219. 407]
[214 412]	[218.09115475 412]
[214 417]	[217.86416862 417]
[219 422]	[218.28121059 422]
[219 427]	[218.57845433 427
[219 432]	[218.80544046 432]
[219 437]	[219. 437]

Gambar 14. Belokan Kelima

4. SIMPULAN

Algoritma pencarian jalur adalah komponen kunci dalam navigasi robot, yang bertujuan untuk menghasilkan jalur yang menghubungkan titik awal (*start point*) dan tujuan (*goal point*) secara efisien. Meskipun algoritma-algoritma ini mampu menentukan jalur yang menghubungkan kedua titik tersebut, sering kali mereka tidak mempertimbangkan karakteristik fisik dari robot, seperti kemampuan untuk menavigasi belokan tajam. Hal ini dapat menyebabkan masalah, terutama ketika robot dihadapkan pada jalur yang memiliki sudut belokan yang tajam, yang bisa mengganggu kestabilan dan efisiensi pergerakan robot.

Penelitian ini berfokus pada penerapan algoritma Gradient Descent untuk melakukan *path smoothing* pada jalur yang telah ditemukan. Dengan menggunakan pendekatan ini, jalur yang memiliki belokan tajam dapat

disesuaikan sehingga menjadi lebih halus dan mudah dilalui oleh robot. Implementasi algoritma ini menunjukkan hasil yang signifikan, di mana belokan-belokan tajam pada belokan pertama sampai belokan kelima pada jalur berhasil dilengkungkan. Hasil ini menunjukkan bahwa Gradient Descent dapat menjadi metode yang efektif untuk meningkatkan kualitas jalur yang dihasilkan oleh algoritma pencarian, dengan memastikan robot dapat menavigasi dengan lebih lancar dan efisien.

Secara keseluruhan, penggunaan algoritma Gradient Descent dalam path smoothing memberikan kontribusi yang signifikan dalam mengatasi masalah navigasi robot di jalur dengan belokan tajam. Simulasi yang dilakukan dengan menggunakan platform V-REP memberikan gambaran yang jelas tentang efektivitas algoritma ini dalam mengoptimalkan jalur yang telah direncanakan. Dengan demikian, penelitian ini menunjukkan bahwa teknik path smoothing berbasis Gradient Descent dapat meningkatkan performa robot, menjadikannya lebih responsif dalam menavigasi jalur yang kompleks dan lebih aman dari potensi kecelakaan atau kesalahan navigasi. Penelitian lebih lanjut dapat mengeksplorasi variasi algoritma optimasi lainnya untuk meningkatkan kinerja *path smoothing* pada berbagai jenis robot dan lingkungan yang lebih dinamis.

DAFTAR PUSTAKA

[1] D. Stmik, "Kendali Logika Fuzzy Pada Robot Line Follower Line Follower Robot with Fuzzy Logic Control," *Citec J.*, vol. 3, no. 1, 2015.

[2] M. Aria, "Algoritma Perencanaan Jalur Kendaraan Otonom berbasis Hibridisasi Algoritma BFS dan Path Smoothing," *Telekontran J. Ilm. Telekomun. Kendali dan Elektron. Terap.*, vol. 8, no. 1, pp. 13–22, 2020, doi: 10.34010/telekontran.v8i1.3083.

[3] D. Richasdy and S. Akbar, "Path Smoothing With Support Vector Regression," *J. Informatics Telecommun. Eng.*, vol. 4, no. 1, pp. 142–150, 2020, doi: 10.31289/jite.v4i1.3856.

[4] S. Osher *et al.*, "Laplacian smoothing gradient descent," *Res. Math. Sci.*, vol. 9, no. 3, pp. 1–28, 2022, doi: 10.1007/s40687-022-00351-1.

[5] S. James, M. Freese, and A. J. Davison, "PyRep: Bringing V-REP to Deep Robot Learning," pp. 1–4, 2019, [Online]. Available: http://arxiv.org/abs/1906.11176.

[6] A. Ravankar, A. A. Ravankar, Y. Kobayashi, Y. Hoshino, and C. C. Peng, "Path smoothing Techniques in Robot Navigation: State-of-The-Art, Current and Future Challenges," *Sensors (Switzerland)*, vol. 18, no. 9, pp. 1–30, 2018, doi: 10.3390/s18093170.

[7] C. L. Bajaj and I. Ihm, "Algebraic Surface Design with Hermite Interpolation," *ACM Trans. Graph.*, vol. 11, no. 1, pp. 61–91, 1992, doi: 10.1145/102377.120081.

[8] L. László, "Cubic Spline Interpolation with Quasiminimal B-spline Coefficients," *Acta Math. Hungarica*, vol. 107, no. 1–2, pp. 77–87, 2005, doi: 10.1007/s10474-005-0180-4.

[9] T. Briand *et al.*, "Theory and Practice of Image B-Spline Interpolation to Cite This Version : HAL Id : hal-01846912 Theory and Practice of Image B-Spline Interpolation Introduction," *Love God Don'T Cite This!*, pp. 99–141, 2018.

[10] M. Freese, F. Ozaki, S. Hirose, and N. Matsuhira, "Virtual Robot Experimentation Platform – A Versatile Small Footprint Robot Simulator," *J. Robot. Mechatronics*, vol. 20, no. 1, pp. 47–60, 2008, doi: 10.20965/jrm.2008.p0047.

[11] S. Azak and E. Erdogan, "Performance Evaluation of the Grid-Based FastSLAM in V-REP Using MATLAB," *14th Int. Conf. Adv. Trends Radioelectron. Telecommun. Comput. Eng. TCSET 2018 - Proc.*, vol. 2018-April, no. April, pp. 276–281, 2018, doi: 10.1109/TCSET.2018.8336202.

[12] H. Batti, C. Ben Jabeur, and H. Seddik, "Mobile Robot Obstacle Avoidance in labyrinth Environment Using Fuzzy Logic Approach," *2019 Int. Conf. Control. Autom. Diagnosis, ICCAD 2019 - Proc.*, no. July, pp. 1–5, 2019, doi: 10.1109/ICCAD46983.2019.9037873.