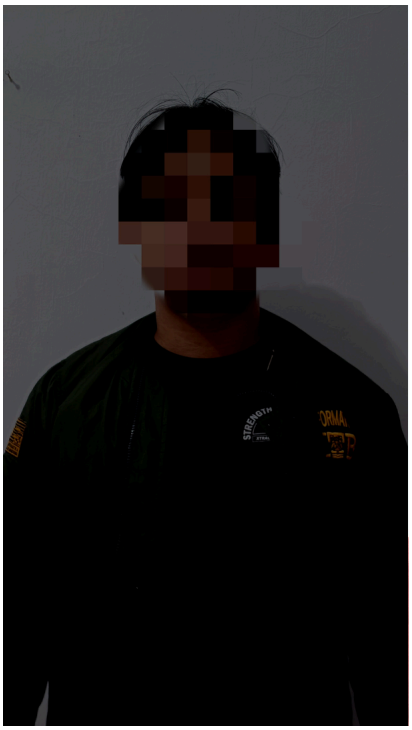
LAPORAN TUGAS BESAR I

IF2211 STRATEGI ALGORITMA




Disusun oleh:

Kelompok Rudal Andriana

Joel Hotlan Haris Siahaan	13523025
Boye Mangaratua Ginting	13523127
Ivant Samuel Silaban	13522129

Program Studi Teknik Informatika

Sekolah Teknik Elektro dan Informatika

Institut Teknologi Bandung

2025

DAFTAR ISI

BAB I
DESKRIPSI TUGAS

Robocode adalah permainan pemrograman yang bertujuan untuk membuat kode bot dalam bentuk tank virtual untuk berkompetisi melawan bot lain di arena. Pertempuran Robocode berlangsung hingga bot-bot bertarung hanya tersisa satu seperti permainan Battle Royale, karena itulah permainan ini dinamakan Tank Royale. Nama Robocode adalah singkatan dari "Robot code," yang berasal dari versi asli/pertama permainan ini. Robocode Tank Royale adalah evolusi/versi berikutnya dari permainan ini, di mana bot dapat berpartisipasi melalui Internet/jaringan. Dalam permainan ini, pemain berperan sebagai programmer bot dan tidak memiliki kendali langsung atas permainan. Pemain hanya bertugas untuk membuat program yang menentukan logika atau "otak" bot. Program yang dibuat akan berisi instruksi tentang cara bot bergerak, mendeteksi bot lawan, menembakkan senjatanya, serta bagaimana bot bereaksi terhadap berbagai kejadian selama pertempuran. Pada Tugas Besar pertama Strategi Algoritma ini, mahasiswa diminta untuk membuat sebuah bot yang nantinya akan dipertandingkan satu sama lain. Tentunya mahasiswa harus menggunakan strategi greedy dalam membuat bot ini. Komponen-komponen dari permainan ini antara lain:

1. Rounds dan Turns

Pertempuran dapat terdiri dari beberapa rounds. Secara default, satu pertempuran berisi 10 rounds, di mana setiap rounds akan memiliki pemenang dan yang kalah. Setiap round dibagi menjadi beberapa turns, yang merupakan unit waktu terkecil. Satu turn adalah satu ketukan waktu dan satu putaran permainan. Jumlah turn dalam satu round tergantung pada berapa lama waktu yang dibutuhkan hingga hanya tersisa bot erakhir yang bertahan.

Pada setiap turn, sebuah bot dapat:

- Menggerakkan bot, memindai musuh, dan menembakkan senjata.
- Bereaksi terhadap peristiwa seperti saat bot terkena peluru atau bertabrakan dengan bot lain atau dinding.
- Perintah untuk bergerak, berputar, memindai, menembak, dan sebagainya dikirim ke server untuk setiap turn.

Perlu diperhatikan bahwa API (Application Programming Interface) bot resmi secara otomatis mengirimkan niat bot ke server di balik layar, sehingga Anda tidak perlu mengkhawatirkannya, kecuali jika Anda membuat API Bot sendiri. Pada setiap turn, bot akan secara otomatis menerima informasi terbaru tentang posisinya dan orientasinya di medan perang. Bot juga akan mendapatkan informasi tentang bot musuh ketika mereka terdeteksi oleh pemindai. Perlu diketahui bahwa game engine yang akan digunakan pada tugas besar ini tidak mengikuti aturan default mengenai komponen Round & Turns.

2. Batas Waktu Giliran

Penting untuk dicatat bahwa setiap bot memiliki batas waktu untuk setiap turn yang disebut turn timeout, biasanya antara 30-50 ms (dapat diatur sebagai aturan pertempuran). Ini berarti bahwa bot tidak bisa mengambil waktu sebanyak yang mereka inginkan untuk bergerak dan menyelesaikan turn saat ini. Setiap kali turn baru dimulai, penghitung waktu ulang diatur ulang dan mulai berjalan. Jika batas waktu tercapai dan bot tidak mengirimkan pergerakannya untuk turn tersebut, maka tidak ada

perintah yang dikirim ke server. Akibatnya, bot akan melewatkan turn tersebut. Jika bot melewatkan turn, ia tidak akan bisa menyesuaikan gerakannya atau menembakkan senjatanya karena server tidak menerima perintah tepat waktu sebelum turn berikutnya dimulai.

3. Energi

Energi Semua bot memulai permainan dengan jumlah energi awal sebanyak 100 poin energi.

- Bot akan kehilangan energi jika ditembak atau ditabrak oleh bot musuh.
- Bot juga akan kehilangan energi jika menembakkan meriamnya.
- Bot akan mendapatkan energi jika peluru dari meriamnya mengenai musuh.

Energi yang didapat akan lebih banyak 3 kali lipat dari energi yang digunakan untuk menembakkan peluru.

- Bot dengan energi nol akan dinonaktifkan dan tidak bisa bergerak. Jika bot terkena serangan dalam keadaan ini, bot akan hancur.

4. Peluru

Semakin banyak energi (daya tembak) yang digunakan untuk menembakkan peluru, semakin berat peluru tersebut dan semakin lambat gerakannya. Namun, peluru yang lebih berat juga menghasilkan lebih banyak kerusakan dan memungkinkan bot mendapatkan lebih banyak energi saat mengenai bot musuh. Seperti disebutkan sebelumnya, peluru yang lebih berat akan bergerak lebih lambat. Ini berarti akan membutuhkan waktu lebih lama untuk mencapai target, meningkatkan risiko peluru tidak mengenai sasaran. Sebaliknya, peluru yang lebih ringan bergerak lebih cepat, sehingga lebih mudah mengenai target, tetapi peluru ringan tidak memberikan banyak poin energi saat mengenai bot musuh.

5. Panas Meriam (Gun Heat)

Saat menembakkan peluru, meriam akan menjadi panas. Peluru yang lebih berat menghasilkan lebih banyak panas dibandingkan peluru yang lebih ringan. Ketika meriam terlalu panas, bot tidak dapat menembak hingga suhu meriam turun ke nol. Selain itu, meriam juga sudah dalam keadaan panas di awal round dan perlu waktu untuk mendingin sebelum bisa digunakan untuk pertama kalinya.

6. Tabrakan

Perlu diperhatikan bahwa bot akan menerima kerusakan jika menabrak dinding (batas arena), yang disebut wall damage. Hal yang sama juga terjadi jika bot bertabrakan dengan bot lain. Jika bot menabrak bot musuh dengan bergerak maju, ini disebut ramming (menabrak dengan sengaja), yang akan memberikan sedikit skor tambahan bagi bot yang menyerang.

7. Bagian Tubuh Tank

Tubuh tank terdiri dari 3 bagian:

- Body adalah bagian utama dari tank yang digunakan untuk menggerakkan tank.
- Gun digunakan untuk menembakkan peluru dan dapat berputar bersama body atau independen dari body.
- Radar digunakan untuk memindai posisi musuh dan dapat berputar bersama body atau independen dari body.

8. Pergerakan

Bot dapat bergerak maju dan mundur hingga kecepatan maksimum. Dibutuhkan beberapa giliran untuk mencapai kecepatan maksimum. Bot dapat mengalami percepatan maksimum sebesar 1 unit per giliran dan pengereman dengan perlambatan maksimum 2 unit per giliran. Percepatan dan perlambatan maksimum tidak bergantung pada kecepatan bot saat itu.

9. Berbelok

Seperti yang disebutkan sebelumnya, bagian tubuh, turret (meriam), dan radar dapat berputar secara independen satu sama lain. Jika turret atau radar tidak diputar, maka keduanya akan mengarah ke arah yang sama dengan tubuh bot. Setiap bagian tubuh memiliki kecepatan putar yang berbeda. Radar adalah bagian tercepat dan dapat berputar hingga 45 derajat per giliran, yang berarti dapat berputar 360 derajat dalam 8 giliran. Turret dan meriam dapat berputar hingga 20 derajat per giliran. Bagian paling lambat adalah tubuh tank, yang dalam kondisi terbaik dapat berputar hingga 10 derajat per giliran. Namun, ini bergantung pada kecepatan bot saat ini. Semakin cepat bot bergerak, semakin lambat kemampuannya untuk berbelok. Perlu diperhatikan bahwa tidak ada energi yang dikonsumsi saat bot bergerak atau berbelok.

10. Pemindaian

Aspek penting dalam Robocode adalah memindai bot musuh menggunakan radar. Radar dapat mendeteksi bot dalam jangkauan hingga 1200 piksel. Musuh yang berada lebih dari 1200 piksel dari bot tidak dapat terdeteksi atau dipindai oleh radar. Penting untuk diperhatikan bahwa sebuah bot hanya dapat memindai bot musuh yang berada dalam jangkauan sudut pemindaian (scan arc)-nya. Sudut pemindaian ini merupakan "sapuan radar" dari arah radar sebelumnya ke arah radar saat ini dalam satu giliran.

Jika radar tidak bergerak dalam suatu giliran, artinya radar tetap mengarah ke arah yang sama seperti pada giliran sebelumnya, maka sudut pemindaian akan menjadi nol derajat, dan bot tidak akan dapat mendeteksi musuh. Oleh karena itu, sangat disarankan untuk selalu mengubah arah radar agar tetap dapat memindai musuh.

11. Skor

Pada akhir pertempuran, setiap bot akan diranking berdasarkan total skor yang diperoleh masing-masing bot selama keseluruhan pertempuran. Tentunya, tujuan utama pada tugas besar ini adalah membuat bot yang memberikan skor setinggi mungkin.

Berikut adalah rincian komponen skor pada pertempuran:

- Bullet Damage: Bot mendapatkan poin sebesar damage yang dibuat kepada bot musuh

menggunakan peluru.

- Bullet Damage Bonus: Apabila peluru berhasil membunuh bot musuh, bot mendapatkan poin sebesar 20% dari damage yang dibuat kepada musuh yang terbunuh.
- Survival Score: Setiap ada bot yang mati, bot lainya yang masih bertahan pada ronde tersebut mendapatkan 50 poin.
- Last Survival Bonus: Bot terakhir yang bertahan pada suatu ronde akan mendapatkan 10 poin dikali dengan banyaknya musuh.
- Ram Damage: Bot mendapatkan poin sebesar 2 kalinya damage yang dibuat kepada bot musuh dengan cara menabrak.
- Ram Damage Bonus: Apabila musuh terbunuh dengan cara ditabrak, bot mendapatkan poin sebesar 30% dari damage yang dibuat kepada musuh yang terbunuh.

Skor akhir bot adalah akumulasi dari 6 komponen diatas. Perlu diperhatikan bahwa game akan menampilkan berapa kali suatu bot meraih peringkat 1, 2, atau 3 pada setiap ronde. Namun, hal ini tidak dihitung sebagai komponen skor maupun untuk perangkingan akhir. Bot yang dianggap menang pertempuran adalah bot dengan akumulasi skor tertinggi.

BAB II
LANDASAN TEORI

2.1 Algoritma Greedy

Algoritma greedy adalah salah satu pendekatan algoritma dalam pemecahan masalah yang bekerja dengan memilih solusi lokal terbaik pada setiap langkah, dengan harapan bahwa solusi ini akan mengarah pada solusi global yang optimal. Algoritma ini tidak selalu menghasilkan solusi yang optimal, tetapi sering memberikan hasil yang cukup baik dalam waktu yang lebih singkat terutama jika dibandingkan dengan pendekatan brute force.

Ciri utama algoritma greedy adalah pemilihan lokal terbaik dimana pada setiap langkah, algoritma memilih pilihan yang terbaik tanpa mempertimbangkan konsekuensi jangka panjang. Ciri lainnya adalah tanpa backtracking dan lebih efisien dalam waktu karena hanya mempertimbangkah langkah saat ini.

2.2 Elemen-Elemen Algoritma Greedy

Algoritma greedy terdiri dari beberapa komponen, yaitu:

1. Himpunan kandidat (C)
   Himpunan yang berisi kandidat yang akan dipilih pada setiap langkah.
   Misal: simpul/sisi di dalam graf, job, task, koin, benda, karakter, dsb.

2. Himpunan solusi (S)
   Himpunan yang berisi kandidat yang sudah dipilih.

3. Fungsi solusi
   Fungsi yang menentukan apakah himpunan kandidat yang dipilih sudah memberikan solusi.

4. Fungsi seleksi (selection function)
   Fungsi yang memilih kandidat berdasarkan strategi greedy tertentu.
   Strategi greedy ini bersifat heuristik.

5. Fungsi kelayakan (feasible)
   Fungsi yang memeriksa apakah kandidat yang dipilih dapat dimasukkan ke dalam himpunan solusi (layak atau tidak).

6. Fungsi objektif:
   Fungsi ini menunjukan sifat memaksimumkan atau meminimumkan.

2.3 Cara Kerja Program

2.3.1 Bagaimana Bot melakukan aksinya

Bot dalam Robocode Tank Royale beroperasi berdasarkan siklus permainan, di mana setiap giliran (turn), bot dapat melakukan:

- Bergerak maju atau mundur
- Memutar badan tank, meriam, atau radar
- Menembak peluru dengan energi tertentu
- Memindai dengan radar untuk mendeteksi keberadaan lawan
- Bereaksi terhadap event seperti tertembak, bertabrakan dengan tembok atau bot lain dll

2.3.2 Mengimplementasikan Algoritma Greedy dalam Bot

Implementasi Algoritma Greedy dalam bot menentukan pemilihan hal yang akan dilakukan bot pada setiap turn. Beberapa aspek yang dapat dipertimbangkan adalah:

- Memilih target yang paling mudah ditembak
- Menghindari serangan musuh dengan pergerak optimal
- Menyesuaikan daya tembak
- Menggunakan pemindaian radar secara efektif

2.3.3 Cara Menjalankan Bot

2.3.3.1 Persiapan dan setup game engine

1. Starter Pack tersedia pada link berikut:

   https://github.com/Ariel-HS/tubes1-if2211-starter-pack

2. Download jar “robocode-tankroyale-gui-0.30.0.jar”, yang merupakan game engine hasil modifikasi asisten.
3. Download dan Ekstrak “TemplateBot.zip” sebagai template bagi bot C#.
4. (opsional) Anda dapat pula mendownload dan ekstrak source code yang berisi:

   ● Sample Bots yang disediakan Robocode.

   ● Source Code game engine yang dapat di build ulang menggunakan gradle apabila dibutuhkan.

2.3.3.2 Cara menjalankan game engine

a. Jalankan file .jar aplikasi GUI
b. Setup konfigurasi booter
c. Jalankan sebuah battle
d. Boot bot yang ingin anda mainkan
e. Tambahkan bot ke dalam permainan

2.3.3.3 Cara mengimplementasikan bot

a. Buatlah folder baru yang bernama bot anda
b. Buatlah file json dengan fields sebagai berikut:

   { "name": "«nama kelompok»", "version": "1.0", "authors": ["«anggota 1»", "«anggota 2»", "«anggota 3»"], "description": "Deskripsi strategi greedy yang digunakan", "homepage": "«…»", "countryCodes": ["«enter your country code, e.g. id»"], "platform": "«enter programming platform, e.g. Java or .Net»", "programmingLang": "C#" }

c. Buatlah file source code C# untuk bot Anda yang bernama bot Anda (e.g. “TemplateBot.cs”)
d. Susun source code bot Anda dengan kerangka minimal sebagai berikut:

```
using Robocode.TankRoyale.BotApi;
using Robocode.TankRoyale.BotApi.Events;
public class BotTemplate : Bot
{
   // The main method starts our bot
   static void Main(string[] args)
   {
      new BotTemplate().Start();
   }

   // Constructor, which loads the bot config file
   BotTemplate() : base(BotInfo.FromFile("BotTemplate.json")) { }

   // Called when a new round is started -> initialize and do some movement
   public override void Run()
   {

      BodyColor = Color.FromArgb(0x00, 0x00, 0x00);
      TurretColor = Color.FromArgb(0x00, 0x00, 0x00);
      RadarColor = Color.FromArgb(0x00, 0x00, 0x00);
      BulletColor = Color.FromArgb(0x00, 0x00, 0x00);
      ScanColor = Color.FromArgb(0x00, 0x00, 0x00);
      TracksColor = Color.FromArgb(0x00, 0x00, 0x00);
      GunColor = Color.FromArgb(0x00, 0x00, 0x00);

      // Repeat while the bot is running
      while (IsRunning)
      {
```

```
            // Write your bot logic
        }
    }
```

e. Buatlah file .csproj yang bernama bot Anda (e.g. “BotTemplate.csproj”). File .csproj harus memiliki isi sebagai berikut:

```
<Project Sdk="Microsoft.NET.Sdk">
  <PropertyGroup>
    <RootNamespace>BotTemplate</RootNamespace>
    <OutputType>Exe</OutputType>
    <TargetFramework>net6.0</TargetFramework>
    <LangVersion>10.0</LangVersion>
  </PropertyGroup>
  <ItemGroup>
    <PackageReference Include="Robocode.TankRoyale.BotApi"
Version="0.30.0"/>
  </ItemGroup>
</Project>
```

f. Buatlah file .cmd dan .sh yang bernama bot Anda (e.g. “BotTemplate.cmd” dan “BotTemplate.sh”). Contoh isi file .cmd dan .sh sebagai berikut:

BotTemplate.cmd

```
@echo off
REM TemplateBot.cmd - Run the bot in development or release mode
REM Set MODE=dev for development (default, always rebuilds)
REM Set MODE=release for release (only runs if bin exists)

set MODE=dev

if "%MODE%"=="dev" (
    REM Development mode: always clean, build, and run
    rmdir /s /q bin obj >nul 2>&1
    dotnet build >nul
    dotnet run --no-build >nul
) else if "%MODE%"=="release" (
    REM Release mode: no rebuild if bin exists
    if exist bin\ (
        dotnet run --no-build >nul
    ) else (
        dotnet build >nul
        dotnet run --no-build >nul
    )
) else (
    echo Error: Invalid MODE value. Use "dev" or "release".
)
```

BotTemplate.sh

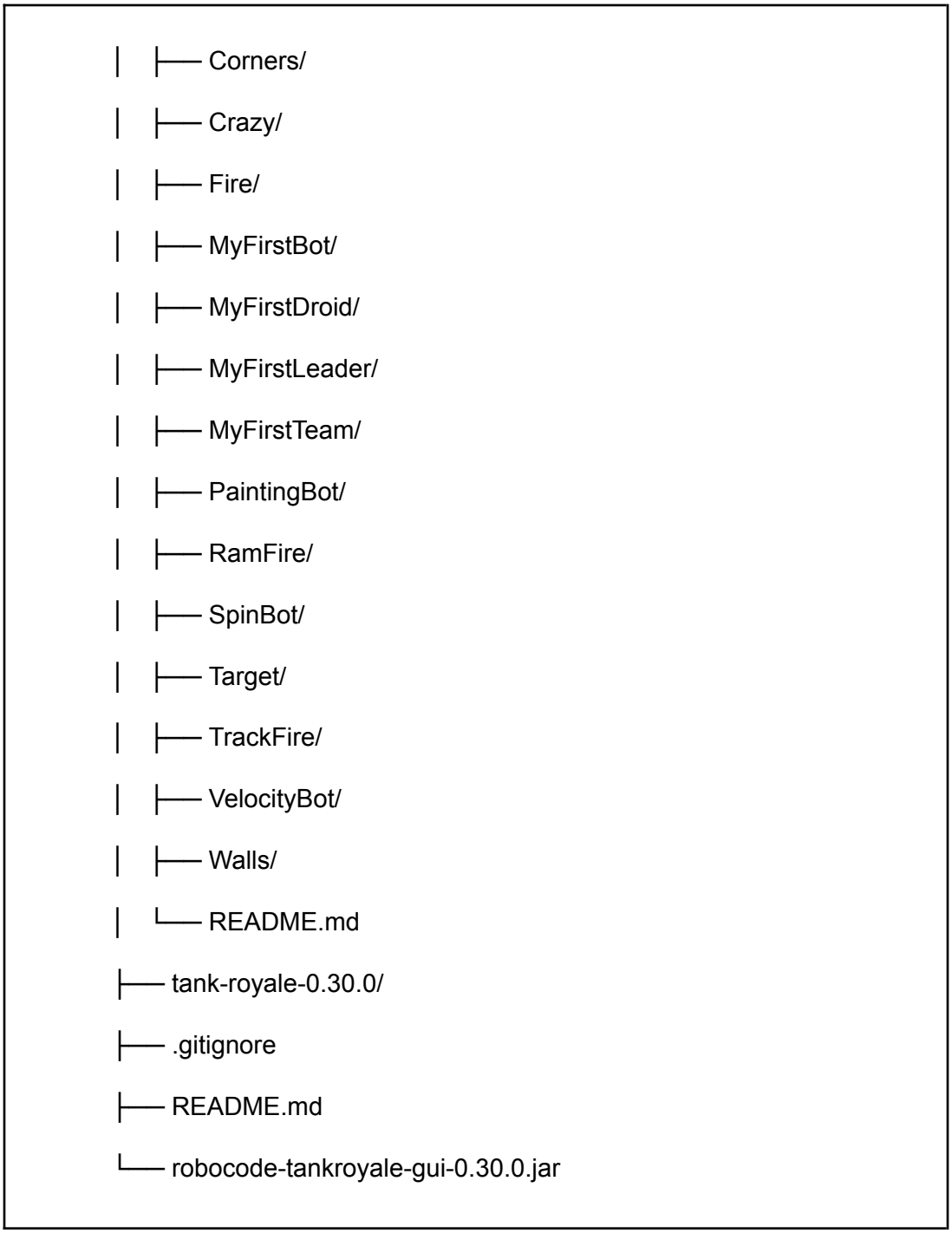
```
#!/bin/sh
# This script runs the bot in development mode by default.
# Development Mode (default): Always cleans, rebuilds, and runs.
# Release Mode (commented out): Runs without rebuilding.

# Development mode: always rebuild
rm -rf bin obj
dotnet build
dotnet run --no-build

# Uncomment below for release mode (runs without rebuilding)
# if [ -d "bin" ]; then
#   dotnet run --no-build
# else
#   dotnet build
#   dotnet run --no-build
# fi
```

2.4.2 Mengembangkan Bot

Pengembangan bot dimulai dengan *starter-pack* v1.0.1 yang dapat ditemukan pada tautan https://github.com/Ariel-HS/tubes1-if2211-starter-pack/releases/tag/v1.0. Struktur data starter pack tersebut adalah sebagai berikut:

Struktur data *tubes1-if2211-starter-pack*

```
tubes1-if2211-starter-pack/
├── TemplateBot/
│   ├── TemplateBot.cmd
│   ├── TemplateBot.cs
│   ├── TemplateBot.csproj
│   ├── TemplateBot.json
│   ├── TemplateBot.sh
├── sample-bots-csharp-0.30.0/
```



Pengembangan bot dilakukan dengan menyalin tubes1-if2211-starter-pack-1.0/TemplateBot. Ubah nama setiap file agar sesuai dengan nama bot yang akan dibuat, kemudian implementasikan strategi greedy pada file cs.

BAB III
APLIKASI STRATEGI GREEDY

3.1 Mapping Persoalan Robocode

Algoritma greedy memiliki beberapa elemen atau istilah yang sering menyertainya, diantaranya:

1. Himpunan kandidat (C): berisi kandidat yang akan dipilih pada setiap langkah.
2. Himpunan solusi (S): berisi kandidat yang sudah dipilih.
3. Fungsi solusi: menentukan apakah himpunan kandidat yang dipilih sudah memberikan solusi.
4. Fungsi seleksi (selection function): memilih kandidat berdasarkan strategi greedy tertentu. Strategi greedy ini bersifat heuristik.
5. Fungsi kelayakan (feasible): memeriksa apakah kandidat yang dipilih dapat dimasukkan ke dalam himpunan solusi (layak atau tidak).
6. Fungsi objektif: memaksimumkan atau meminimumkan

Persoalan di dalam Robocode dibagi menjadi *game objects* dan *property* dari *game objects* yang ada, sehingga mapping yang dihasilkan dari persoalan tersebut dapat dirangkum menjadi:

Tabel 3.1.1 Mapping Persoalan Robocode

No	**Persoalan**	**Kegunaan**	**Fungsi Proses**
1	Round dan Turns	- Pada setiap turn, bot dapat memilih untuk melakukan antara bergerak, memindai musuh dan menembakkan senjata. - Setiap suatu ronde selesai, bot yang bertahan hidup mendapatkan poin tambahan.	-
2	Energi	- Energy digunakan untuk menentukan kelangsungan hidup dan menembakan peluru. - Bot yang kehabisan energi akan mati ketika terkena peluru atau *ram* dari bot lain.	-
3	Peluru	- Senjata utama untuk menyerang musuh. - Membutuhkan energi untuk ditembakan. - Penembakan peluru diproses dengan strategy greedy by distance dan	-CalculateFirePower() -SmartFire()

		greedy by enemy’s energy.	

4	Tabrakan	- Sebuah bot dapat melakukan *ram* kepada musuh untuk mendapatkan poin. - Aksi menabrak diproses dengan strategy greedy by distance.	-
5	Bagian Tubuh Tank	- Struktur utama bot (Body, Gun, Radar) - Setiap bagian dapat berotasi secara independen dengan kecepatan berbeda. - Rotasi bagian tubuh disesuaikan dengan strategi.	-
6	Pergerakan	- Suatu rangkaian gerakan untuk melakukan suatu tujuan - Contoh: gerak menuju musuh, gerak menghindari peluru.	-MoveToCEnter() -CariArahAman()
7	Pemindaian	- Mendeteksi musuh dan peluru beserta atribut-atributnya.	-

3.2 Eksplorasi Alternatif Solusi

Menurut Kelompok Kami, terdapat beberapa alternatif solusi baik menggunakan algoritma berbasis *greedy* maupun *non-greedy*. Terdapat beberapa alternatif algoritma yang kami eksplorasi adalah sebagai berikut:

3.2.1 Greedy by energy

Greedy by energy adalah pendekatan algoritma greedy dengan memprioritaskan menembak bot musuh dengan energi paling rendah. Bot akan melakukan pemindaian untuk menemukan bot musuh dengan energi paling rendah. Bot ini akan selalu menembakkan peluru ke bot musuh tersebut dengan kekuatan yang diatur sesuai jarak antara bot musuh dengan bot.

a. Mapping Elemen Greedy

i. Himpunan kandidat : Semua Aksi yang tersedia.
ii. Himpunan solusi : Aksi yang terpilih.
iii. Fungsi solusi : Memeriksa apakah bot musuh memiliki energi paling sedikit.
iv. Fungsi seleksi : Pilih bot yang memiliki energi paling sedikit.

v. Fungsi kelayakan : Memeriksa apakah bot musuh masih memiliki energi paling sedikit.
vi. Fungsi objektif : Menembak mati bot musuh dengan energi terkecil.

b. Analisis Efisiensi Solusi

Aksi utama yang dilakukan oleh bot adalah aksi pemindaian energi bot-bot musuh dan aksi menembak. Bot akan memprioritaskan untuk menembak bot musuh dengan energi terkecil. Bot akan menyimpan data id serta energi bot yang dipindai. Dalam pemindaian berikutnya bot akan memeriksa jika bot yang baru di scan memiliki energi yang lebih rendah dari bot yang sebelumnya. Proses tersebut relatif cepat, namun penembakan peluru masih kurang efisien. Hal tersebut dikarenakan bot menembak musuh setiap satu putaran pemindaian. Ketika bot musuh memiliki jarak yang jauh, penembakan hanya akan menggunakan sedikit energi, sehingga poin yang didapat hanya sedikit. Poin tidak sebanding dengan waktu yang diperlukan untuk melakukan aksi tersebut. Dalam hal ini pemilihan bot musuh dengan energi terkecil tidak sebanding dengan jumlah gerakan dan waktu yang diperlukan untuk mendapatkan poin dari menembak mati bot musuh.

c. Analisis Efektivitas Solusi

Greedy by energy sangat bergantung pada kesempatan menembak mati bot musuh yang memberikan poin besar. Jarak dari musuh juga mempengaruhi kesempatan peluru mengenai bot musuh yang memiliki energi paling sedikit.

Strategi ini efektif jika:

- Luas arena cukup kecil sehingga jarak antar bot relatif kecil.
- Bot yang berada di arena tidak terlalu banyak.

Strategi ini tidak efektif jika:

- Jarak bot musuh dengan energi paling sedikit sangat jauh.
- Bot target terhalangi oleh bot lain yang energinya banyak.

3.2.2 Greedy by Targeting

Greedy by targeting adalah pendekatan algoritma greedy dengan berfokus pada pemilihan target dan penembakan dengan efisiensi yang tinggi. Bot ini akan mengutamakan penguncian dan penembakan musuh dengan akurasi tinggi.

a. Mapping Elemen Greedy

i. Himpunan kandidat : Semua posisi musuh yang terdeteksi radar
ii. Himpunan solusi : Tembakan yang telah dilepaskan ke target yang terdeteksi
iii. Fungsi solusi : Bot terus mengunci target dan menembak dengan akurasi tinggi.
iv. Fungsi seleksi : Memilih dan mengunci target dengan posisi terbaru dan menyesuaikan sudut tembakan untuk akurasi maksimal
v. Fungsi kelayakan: Tembakan hanya dilakukan jika sudut tembak optimal,

senjata sudah siap, dan sudah mengunci musuh.

vi. Fungsi objektif: Memaksimalkan jumlah tembakan target

b. Analisis Efisiensi Solusi

Strategi Greedy by targeting memiliki efisiensi yang tinggi dalam konsumsi energi dan akurasi tembakan. Dengan selalu mengunci target dan menembak saat target sudah terkunci dan sudut optimal tercapai, bot dalam melakukan penembakan terus menerus dan memaksimalkan damage tanpa membuang banyak peluru. Salah satu kunci survive dari bot ini adalah selama bot ini menembak bot lain dan mengenainya, energi bot akan bertambah terus. Namun kekurangan dari bot ini adalah, bot ini lemah jika diadu dengan bot dengan bot lain yang memiliki gerakan yang acak dan cepat.

c. Analisis Efektivitas Solusi

Solusi ini dinilai efektif apabila:

- Efektif melawan bot yang diam atau bergerak dengan pola yang tetap

Solusi ini dinilai kurang efektif apabila:

- Tidak efektif melawan bot dengan pola acak dan cepat
- Untuk beberapa kondisi, bot ini tidak diuntungkan saat mode Free For All.

3.2.3 Greedy by movement

Greedy by movement adalah pendekatan algoritma greedy dengan yang berfokus pada optimalisasi pergerakan bot di arena. Bot yang memanfaatkan strategi ini akan melakukan pergerakan acak dengan heuristic untuk membuat gerakan acaknya lebih efektif, seperti pergerakan yang membuat dirinya sulit mengenai peluru, menghindari tembok arena, gerakan saat menabrak bot lain, dan lain-lain.

a. Mapping Elemen Greedy

i. Himpunan kandidat : Semua kemungkinan posisi yang dapat ditempati bot berdasarkan kecepatan dan arah gerak saat ini.

ii. Himpunan solusi : Posisi yang telah dipilih bot untuk ditempati dalam waktu tertentu.

iii. Fungsi solusi : Bot terus bergerak untuk menghindari tembakan dan mencari posisi strategis.
iv. Fungsi seleksi : Memilih posisi yang paling aman dan memberikan keuntungan dalam pertempuran berdasarkan pola tembakan musuh dan lingkungan sekitar.
v. Fungsi kelayakan : Posisi baru hanya dipilih jika meningkatkan peluang bertahan, tidak membatasi kemampuan menembak, dan sesuai dengan batasan arena.
vi. Fungsi objektif : Memaksimalkan kelangsungan hidup dengan menghindari tembakan dan memperoleh posisi terbaik untuk menembak.

b. Analisis Efisiensi Solusi

Bot yang melakukan pendekatan *greedy by movement* memiliki efisiensi tinggi dalam menghindari tembakan dan mempertahankan energi bot. Dengan selalu bergerak secara optimal, bot dapat mengurangi kemungkinan terkena tembakan sambil mencari celah untuk menyerang. Namun, strategi ini bisa mengorbankan peluang menembak jika terlalu fokus pada pergerakan.

c. Analisis Efektivitas Solusi

Solusi yang menggunakan pendekatan *greedy by movement* memiliki

Solusi ini dinilai efektif apabila:

- Efektif melawan bot yang mengandalkan tembakan langsung dengan prediksi pergerakan biasa.

- Cocok untuk mode *Free For All* di mana banyak bot saling menembak.

Solusi ini dinilai kurang efektif apabila:
- Kurang efektif jika bot lawan memiliki senjata dengan kecepatan tinggi atau pola tembakan yang dapat mengantisipasi gerakan acak.
- Bisa kehilangan peluang menyerang jika terlalu sering berpindah tanpa mempertimbangkan posisi optimal untuk menembak.

3.2.4 Greedy by Fire

Greedy by fire adalah pendekatan algoritma greedy yang mengoptimalkan kekuatan tembakan berdasarkan jarak musuh. Bot ini akan memilih ukuran peluru yang sesuai untuk memaksimalkan peluang mengenai target dan efisiensi energi.

a. Mapping Elemen Greedy

i. Himpunan kandidat : Semua kemungkinan kekuatan tembakan berdasarkan jarak musuh.

ii. Himpunan solusi : Tembakan dengan ukuran yang telah dipilih dan dilepaskan.
iii. Fungsi solusi : Bot terus menyesuaikan ukuran peluru agar sesuai dengan jarak musuh dan peluang tembakan mengenai target.
iv. Fungsi seleksi : Memilih ukuran peluru yang memberikan peluang mengenai target paling tinggi berdasarkan jarak dan kecepatan peluru.
v. Fungsi kelayakan : Tembakan hanya dilakukan jika peluang kena cukup tinggi, energi cukup tersedia, dan posisi musuh bisa diprediksi.
vi. Fungsi objektif : Memaksimalkan jumlah tembakan yang mengenai target dengan efisiensi energi terbaik.

b. Analisis Efisiensi Solusi

Strategi *Greedy by fire* memiliki efisiensi tinggi dalam penggunaan energi dan akurasi tembakan. Dengan memilih peluru kecil untuk target jauh (karena lebih cepat) dan peluru besar untuk target dekat (karena lebih kuat), bot dapat memaksimalkan damage tanpa membuang energi. Namun, strategi ini bisa kurang efektif jika bot lawan bergerak tidak terduga atau memiliki pergerakan zig-zag yang sulit diprediksi.

c. Analisis Efektivitas Solusi

Solusi ini dinilai efektif apabila:
- Efektif melawan bot dengan pergerakan linear atau berpola, karena peluang tembakan mengenai target lebih tinggi.
- Sangat efisien dalam mode *1v1*, di mana energi bot dapat dimanfaatkan dengan baik.

Solusi ini dinilai kurang efektif apabila:
- Kurang efektif melawan bot dengan pola gerak acak dan cepat.
- Dalam mode *Free For All*, bisa kurang optimal karena sering terjadi perubahan target sebelum peluru mengenai sasaran.

3.3 Faktor Pengaruh Efektivitas Secara Umum

Terdapat beberapa faktor eksternal yang dapat mempengaruhi performa semua bot, terlepas dari algoritma yang digunakan. Faktor-faktor tersebut adalah sebagai berikut:

1. Posisi Awal Bot dalam Arena.

   Posisi Awal bot sangat mempengaruhi strategi pergerakan, seperti bot yang berada di sudut arena memiliki keterbatasan ruang gerak dibandingkan bot yang berada di tengah. Ada juga kondisi dimana banyak bot memiliki posisi yang berdekatan satu sama lain, bahkan sampai ada bot yang malah dikepung bot lain, hal itu sangat mempengaruhi efektivitas bot.

2. Penyebaran Musuh dalam Arena

   Untuk beberapa bot, jika musuh tersebar secara acak (biasanya kondisi ini terjadi jika sisa bot tersisa hanya sedikit), maka bot perlu melakukan pencarian aktif yang lebih kompleks yang dapat mempengaruhi efisiensi strategi greedy.

3. Keberuntungan dalam Dinamika Pertarungan

   Meskipun algoritma greedy dapat membantu bot dalam mengambil keputusan terbaik, ada faktor keberuntungan lain yang berperan seperti:

   - Posisi awal musuh yang menguntungkan bagi bot tertentu
   - Peluru yang tidak sengaja mengenai musuh yang tidak sekarat.
   - Situasi dimana dua bot bertarung hingga salah satu kalah, sementara bot lain hanya perlu menyelesaikan lawan yang sudah lemah.
   - Bot yang mengunci lawan, namun ketika menembak malah mengenai bot lain.

Dari faktor-faktor tersebut, dapat disimpulkan bahwa meskipun strategi greedy berusaha mengoptimalkan performa bot, masih ada beberapa aspek yang dipengaruhi faktor keberuntungan, yang dapat menentukan hasil akhir pertarungan.

3.4 Strategi Greedy yang Dipilih

Strategi yang pada akhirnya digunakan dalam implementasi bot ini adalah **Greedy by Energy, Targeting, Movement, dan Fire**. Pemilihan strategi ini didasarkan pada keseimbangan antara agresivitas dan efisiensi dalam pertempuran. *Greedy by Energy* memastikan bahwa bot selalu berusaha mempertahankan atau meningkatkan energinya dengan cara yang optimal, seperti menghindari tembakan lawan dan menyerang secara efektif untuk mendapatkan tambahan energi dari serangan yang berhasil. *Greedy by Targeting* memungkinkan bot untuk selalu fokus pada musuh yang paling mudah ditembak, baik dari segi posisi maupun pola pergerakan, sehingga bot tidak membuang banyak peluru secara sia-sia. Sementara itu, *Greedy by Movement* digunakan untuk mengoptimalkan pergerakan bot agar dapat menghindari serangan musuh sekaligus mencari posisi terbaik untuk menyerang. *Greedy by Fire* membantu bot dalam menyesuaikan daya tembak berdasarkan jarak musuh, di mana peluru dengan daya lebih kecil digunakan untuk target yang lebih jauh agar memperbesar kemungkinan mengenai sasaran.

Dengan menggabungkan keempat strategi *Greedy* ini, bot yang diimplementasikan menjadi lebih

adaptif terhadap berbagai situasi dalam pertempuran. Bot dapat mempertahankan daya tahan lebih lama dengan pengelolaan energi yang baik, sekaligus tetap agresif dalam menyerang musuh yang paling menguntungkan. Selain itu, bot juga mampu menghindari situasi berbahaya dengan pergerakan yang optimal, sehingga tidak mudah terkena serangan lawan. Pendekatan ini memberikan keseimbangan antara ofensif dan defensif, menjadikannya strategi yang efektif untuk menghadapi berbagai jenis lawan, baik yang pasif maupun agresif. Dengan mempertimbangkan efektivitas dari masing-masing strategi, kombinasi ini dipilih agar bot dapat bertahan lebih lama di medan pertempuran dan memiliki peluang kemenangan yang lebih tinggi dibandingkan jika hanya menggunakan satu strategi secara terpisah.

Kami membuat empat buah bot yang merupakan kombinasi greedy-greedy sebelumnya. Dari bot utama yang merupakan hasil implementasi strategi Greedy by Movement and Fire, lalu ada bot implementasi greedy by Energy dan Fire, Targeting dan Fire

Untuk bot utama (Rudal_Andriana_Main) yang merupakan implementasi strategi greedy by movement and fire, dimana bot akan berjalan berputar layaknya spin bot dengan radius berubah-ubah, dan akan mengubah arah rotasi jika mengenai dinding ataupun bot lain. Dan dari greedy by fire, bot akan menembak sesuai dengan jarak dan energi musuh yang dipindai. Jika musuh yang dipindai sangat dekat dan/atau energi musuh sangat kecil, dia akan menembak dengan kekuatan penuh. Jika jarak musuh semakin jauh, maka energi tembakan akan semakin kecil, dengan pertimbangan peluru semakin cepat meluncur dan lebih hemat energi (musuh yang jauh sangat sulit dikenai tembakan, sehingga tidak perlu energi yang besar untuk ditembakkan. pertimbangan lainnya ialah agar peluru semakin cepat melesat dan memperbesar peluang mengenai target).

Sedangkan untuk bot alternatif 1 (Rudal_Andriana_1) yang merupakan implementasi dari strategi energy and fire, dimana bot akan berjalan zig zag, dan ketika musuh terdeteksi oleh radar, bot akan menghadap dan bergerak ke arah musuh sambil menembak sesuai jarak. Jika ditabrak atau ram dari belakang akan maju dan berputar ke kanan, selain itu akan mundur memutar ke kanan. Dan karena bot ini bergerak mengikuti musuh, maka bot ini akan spam tembakan.

Sedangkan untuk bot alternatif 2 (Rudal_Andriana_2) hanya akan bergerak maju mundur sambil memutar gun dan radar 360 derajat, dan jika ada musuh terdeteksi oleh radar, maka akan dicatat energy dan id nya, dan jika energinya lebih kecil dari energy target yg disimpan bakal ditembak dan id dan energi simpanannya bakal diperbarui. Bot ini juga menggunakan greedy dalam menembak musuh, dimana semakin dekat maka energi yang digunakan untuk menembak akan semakin besar.

Sedangkan untuk bot alternatif 3 (Rudal_Andriana_3) yang merupakan implementasi dari strategi greedy by Targeting dan Fire, bot akan bergerak secara acak, namun berusaha bergerak ke arah tengah arena. Namun, ketika ada bot yang terdeteksi, maka bot akan lebih berfokus dalam menembak bot yang terdeteksi hingga bot target bergerak keluar dari daerah pandangan radar bot tersebut.

BAB IV
IMPLEMENTASI DAN PENGUJIAN

4.1. Implementasi Bot Greedy

4.1.1 Bot Rudal_Andriana_Main

Pseudocode:

```
WHILE bot is running:
    Set warna bot: (Black, Black, Black, Yellow)

   // Hitung jarak ke dinding terdekat
    jarakKeDinding = HitungJarakKeDinding()

   // Jika terlalu dekat dengan dinding, cari arah aman
    IF jarakKeDinding < 200 THEN
       arahAman = CariArahAman()
       SetTurnRight(arahSetelahTabrakDinding * arahAman)
       Forward(150)
       Go()
    ELSE
       // Pergerakan normal dengan radius dinamis
       jariJari += arahPerubahanJariJari

      IF jariJari > jariJariMaksimum OR jariJari < jariJariMinimum THEN
          arahPerubahanJariJari = -arahPerubahanJariJari

      kecepatanTarget = 6 + (jariJari / jariJariMaksimum * 3.5)
       SetTurnRight(arahSetelahTabrakDinding * 5000)
       Forward(jariJari)
       Go()



On ScannedBotEvent:

// Jika bot melihat musuh, hitung jarak dan energi
 jarak = DistanceTo(enemyX, enemyY)
 kekuatanTembak = HitungKekuatanTembak(jarak, enemyEnergy, botEnergy)

Fire(kekuatanTembak)

On HitBotEvent:

// Jika menabrak bot lain, mundur dan berputar
 SetTurnLeft(45)
 Back(80)
```

```
 Go()

On HitWallEvent:

// Jika menabrak dinding, ubah arah dan mundur
 IF X > (ArenaWidth - 7) OR Y < 7 THEN
    arahSetelahTabrakDinding = -1
 ELSE
    arahSetelahTabrakDinding = 1

SetTurnRight(arahSetelahTabrakDinding * 135)
 Back(100)
 Go()

HitungJarakKeDinding:

jarakX = MIN(X, ArenaWidth - X)

   jarakY = MIN(Y, ArenaHeight - Y)

   RETURN MIN(jarakX, jarakY)

CariArahAman:

arahKePusat = DirectionTo(ArenaWidth / 2, ArenaHeight / 2)

   RETURN arahKePusat


HitungKekuatanTembak(jarak, energiTarget, energiSendiri):

IF jarak < 100 AND energiTarget < 20 THEN

      RETURN 3 // Kekuatan maksimum

   ELSE IF jarak < 50 THEN

      RETURN 3 // Kekuatan maksimum

   ELSE IF jarak < 400 THEN

      RETURN 2 // Kekuatan sedang

   ELSE

      RETURN 1 // Kekuatan rendah


Struktur Data:
```

-

Fungsi:
- HitungKekuatanTembak(jarak, energiTarget, energiSendiri)

Prosedur:
- CariArahAman()
- HitungJarakKeDinding()
- Run()
- OnScannedBot()
- OnHitBot()
- OnHitWall()

4.1.2 Bot Rudal_Andriana_1

Pseudocode:

```
WHILE bot is running:
   // Greedy Movement: Move zig-zag
   SetTurnLeft(400)
   Forward(200)
   SetTurnRight(400)
   Forward(200)

ON ScannedBotEvent:
   // Greedy Firing: Fire at the closer bot with higher power
   IF (DistanceTo(enemeyBot ) < 100 AND DistanceTo(enemyBot) >= 0) THEN :
      Fire(3)
   ELSE IF (DistanceTo(enemyBot) < 200 AND DistanceTo(enemyBot) >= 100) THEN:
      Fire(2.5)
   else :
      Fire(1)

   // Greedy Movement: Move towards the scanned bot
   FaceTarget(enemyBot);
   Forward(10);

ON HitBotEvent:
   // Greedy Evasion : move away from the hit bot
   var bearing = GunBearingTo(enemyBot)
   IF (bearing >= 140 OR bearing <= -140) THEN:
      SetTurnRight(80)
      Forward(80)
   ELSE :
      FaceTarget(enemyBot)
      SetTurnRight(80)
      Back(80)

ON HitWallEvent:
```

```
    // Greedy Wall Avoidance: Move away from the wall
    TurnLeft(40)
    Forward(80)
```

Struktur Data:
-

Fungsi:
- DistanceTo(enemyX, enemyY)
- GunBearingTo(enemyX, enemyY)

Prosedur:
- Run()
- FaceTarget(double x, double y)
- OnScannedBot(ScannedBotEvent e)\
- OnHitWall(HitWallEvent e)
- OnHitBot(HitBotEvent e)
- OnWonRound(WOnRoundEvent e)

4.1.3 Bot Rudal_Andriana_2

Pseudocode:

```
WHILE bot is running:
    // Scan arena untuk mendeteksi bot musuh
    TurnGunLeft
    times++

    // Jika tidak ada tembakan selama tiga kali pemindaian, set ulang bot target
    IF (times >= 3) THEN :
        targetId= -1
        targetEnergy = 999
        times = 0

    Forward(100)

ON ScannedBotEvent:
    // Menyimpan id dan energi bot musuh jika energi lebih kecil atau belum ada target bot
    double energy = e.Energy
    int id = e.ScannedBotId
    var distance = DistanceTo(e.X, e.Y)

    IF (targetId == -1 OR energy < targetEnergy) THEN:
        targetId = id
        targetEnergy = energy
        SmartFire(distance)

SmartFire :
```

```
    IF (distance < 150 AND distance >= 0) THEN :
        Fire(3)
    ELSE IF (distance < 400 AND distance >= 150) THEN:
        Fire(2)
    ELSE :
        Fire(1)

ON BotDeathEvent:
    // Reset target bot
    IF (e.VictimId = targetId) THEN:
        targetId = -1
        targetEnergy = -999
        times = 0

ON HitWallEvent:
    // Turn away from wall
    TurnRight(180)


ON HitBotEvent:
    // Turn away from wall
    TurnRight(90)

ON BulletFiredEvent:
    times- -
```

Struktur Data:
-

Fungsi:
- DistanceTo(enemyX, enemyY)

Prosedur:
- Run()
- SmartFire()
- OnScannedBot(ScannedBotEvent e)\
- OnBotDeath(BotDeathEvent e)
- OnBulletFired(BulletFiredEvent e)
- OnHitWall(HitWallEvent e)
- OnHitBot(HitBotEvent e)

4.1.4 Bot Rudal_Andriana_3

Pseudocode:

```
WHILE bot is running:
    // Greedy Movement: Move toward the center to avoid walls
    MoveToCenter()
```

```
    // Greedy Targeting: Continuously scan for enemies
    TurnGunLeft(10)

ON ScannedBotEvent:
    // Greedy Targeting: Aim at the scanned enemy
    bearingFromGun = GunBearingTo(enemyX, enemyY)
    TurnGunLeft(bearingFromGun)

    // Greedy Firing: Adjust firepower based on distance
    firepower = CalculateFirepower(distance)
    IF gun is cool AND bearingFromGun is small:
        Fire(firepower)

ON HitByBulletEvent:
    // Greedy Evasion: Move away from the bullet's direction
    TurnRight(random angle)
    Back(random distance)
    TurnRight(random angle)
    Forward(random distance)

ON HitWallEvent:
    // Greedy Wall Avoidance: Turn away from the wall
    Back(100)
    TurnRight(random angle)
    Forward(100)

ON HitBotEvent:
    IF enemy is rammed:
        // Greedy Ramming: Fire aggressively
        Fire(3)
    ELSE IF enemy energy is low:
        // Greedy Attack: Finish off weak enemies
        Forward(30)
        Fire(3)
    ELSE:
        // Greedy Retreat: Back off from stronger enemies
        Back(50)
        TurnRight(random angle)
        Forward(50)
```

Struktur Data:
-

Fungsi:

- CalculateFirepower(distance)
- DistanceTo(enemyX, enemyY)
- GunBearingTo(enemyX, enemyY)

- NormalizeBearing(angle)

Prosedur:
- Run()
- MoveToCenter()
- OnScannedBot(ScannedBotEvent e)
- OnHitByBullet(HitByBulletEvent e)
- OnHitWall(HitWallEvent e)
- OnHitBot(HitBotEvent e)

4.2 Analisis dan Pengujian

4.2.1 Pertandingan antar bot

Test 1 :

Rank	Name	Total Score	Survival	Surv. Bonus	Bullet Dmg.	Bullet Bo...	Ram Dmg.	Ram Bonus	1sts	2nds	3rds
1	Rudal_Andriana_Mai...	926	350	60	448	45	23	0	2	0	2
2	Rudal_Andriana_1 1.0	876	350	30	433	20	42	0	2	2	1
3	Rudal_Andriana_3 1.0	727	300	30	356	5	25	10	1	2	0
4	Rudal_Andriana_2 1.0	456	200	0	238	15	2	0	0	1	2

Test 2 :

Rank	Name	Total Score	Survival	Surv. Bonus	Bullet D...	Bullet Bo...	Ram Dmg.	Ram Bonus	1sts	2nds	3rds
1	Rudal_Andriana_Mai...	1163	500	60	502	56	44	0	3	0	2
2	Rudal_Andriana_3 1.0	1077	450	60	454	44	68	0	2	2	0
3	Rudal_Andriana_1 1.0	894	300	0	526	18	50	0	0	3	1
4	Rudal_Andriana_2 1.0	322	100	0	208	4	10	0	0	0	2

Test 3 :

Rank	Name	Total Score	Survival	Surv. Bonus	Bullet D...	Bullet Bo...	Ram Dmg.	Ram Bonus	1sts	2nds	3rds
1	Rudal_Andriana_1 1.0	1013	400	30	472	72	38	0	2	0	1
2	Rudal_Andriana_3 1.0	958	300	0	590	22	46	0	2	2	1
3	Rudal_Andriana_Mai...	828	300	60	418	20	30	0	1	1	2
4	Rudal_Andriana_2 1.0	613	350	30	206	17	10	0	0	2	1

4.2.2 Analisis dari hasil pengujian

Berdasarkan hasil pengujian dari ketiga tes yang dilakukan, dapat dilakukan analisis terhadap efektivitas masing-masing bot dalam pertandingan Robocode.

a) Analisis Bot Utama (Rudal_Andriana_Main)

Bot utama yang menggunakan strategi Greedy by Movement dan Fire menunjukkan performa yang cukup baik dalam pertandingan. Dalam dua dari tiga tes (Test 1 dan Test 2), bot ini berhasil meraih peringkat pertama, dengan skor total yang tinggi dan damage peluru yang besar. Hal ini menunjukkan bahwa strategi pergerakan melingkar dengan radius yang berubah-ubah cukup efektif dalam menghindari serangan lawan serta tetap menjaga posisi yang baik dalam bertarung.

Selain itu, penerapan Greedy by Fire membantu bot dalam menyesuaikan daya tembak berdasarkan jarak dan energi musuh, sehingga penggunaan energi lebih efisien. Namun, dalam Test 3, bot ini mengalami sedikit penurunan performa dan hanya berada di

posisi ketiga. Ini kemungkinan disebabkan oleh perubahan dinamika pertarungan yang membuat strategi pergerakan dan penembakan tidak seefektif di tes sebelumnya.

b) Analisis Bot Alternatif 1 (Rudal_Andriana_1)

Bot ini menerapkan strategi Greedy by Energy dan Fire, dengan pola pergerakan zig-zag serta agresif dalam menyerang musuh. Dari hasil pengujian, bot ini berhasil memenangkan Test 3 dan memperoleh skor tinggi di Test 1 dan Test 2. Pola pergerakan zig-zag membantu dalam menghindari serangan lawan, sementara strategi menembak berdasarkan jarak membuat serangan bot ini cukup efisien.

Kelebihan utama dari bot ini adalah kemampuannya untuk mendekati musuh dan terus menembak secara agresif, yang memberikan damage besar kepada lawan. Namun, karena bot ini cenderung mendekati musuh, ia juga lebih rentan terhadap serangan balasan, terutama dari lawan yang memiliki pola pergerakan tidak terprediksi.

c) Analisis Bot Alternatif 2 (Rudal_Andriana_2)

Bot ini memiliki strategi yang lebih pasif dengan hanya bergerak maju-mundur sambil memindai area sekelilingnya. Dari hasil pengujian, bot ini selalu berada di peringkat terakhir dalam setiap tes, yang menunjukkan bahwa strategi yang digunakan kurang efektif dalam pertempuran kompetitif.

Meskipun bot ini memiliki sistem pemilihan target berdasarkan energi musuh yang lebih kecil, pendekatan pergerakannya yang terbatas membuatnya mudah menjadi sasaran empuk bagi bot lain yang lebih agresif. Selain itu, bot ini kurang memiliki mekanisme penghindaran serangan yang baik, sehingga lebih mudah terkena tembakan lawan.

d) Analisis Bot Alternatif 3 (Rudal_Andriana_3)

Bot ini menggunakan strategi Greedy by Targeting dan Fire, dengan pergerakan acak dan kecenderungan menuju tengah arena. Dari hasil pengujian, bot ini menunjukkan performa yang cukup baik di Test 2, di mana ia berhasil menempati peringkat kedua dengan skor tinggi.

Strategi bot ini cukup agresif karena lebih memprioritaskan menembak musuh yang terdeteksi oleh radar, namun kelemahannya adalah pergerakan yang tidak selalu optimal. Bot ini juga bisa kehilangan target jika musuh bergerak keluar dari jangkauan radar, yang menyebabkan bot membuang waktu dalam menemukan target baru.

Kesimpulan

Dari hasil pengujian, bot dengan strategi Greedy by Movement dan Fire (Rudal_Andriana_Main) serta Greedy by Energy dan Fire (Rudal_Andriana_1) terbukti lebih efektif dalam pertandingan. Bot utama memiliki keseimbangan antara pertahanan dan serangan, sementara bot alternatif 1 lebih agresif dalam menghadapi lawan. Sebaliknya, bot yang terlalu pasif atau tidak memiliki pergerakan optimal (Rudal_Andriana_2 dan 3) cenderung kurang kompetitif.

Dengan demikian, strategi yang mengoptimalkan pergerakan menghindari serangan dan pemilihan daya tembak yang sesuai dengan jarak musuh adalah strategi Greedy yang paling efektif dalam Robocode.

BAB V
KESIMPULAN DAN SARAN

5.1 Kesimpulan

Dari hasil tugas ini, algoritma greedy merupakan metode yang efektif dalam pengambilan keputusan secara cepat berdasarkan situasi saat ini, meskipun belum tentu menghasilkan solusi global paling optimal. Namun, dalam konteks Robocode, pendekatan greedy terbukti memberikan performa yang cukup baik, terutama dalam aspek pergerakan dan penargetan lawan. Dengan strategi ini, bot dapat beradaptasi dengan kondisi pertempuran secara mandiri, meningkatkan peluang bertahan, dan bersaing secara kompetitif dalam pertandingan.

5.2 Saran

Beberapa saran untuk kelompok ini adalah :

1. Memulai pengerjaan tugas lebih awal dan tidak menunda-nunda
2. Mendalami lebih lanjut terkait konsep algoritma greedy, terutama dalam konteks pemrograman bot dan strategi permainan.
3. Memperbanyak eksplorasi mengenai ilmu yang dibutuhkan dalam mengerjakan tugas

LAMPIRAN

Link Repository: jandhiesto/Tubes1_rudal-andriana

Link Rilis Github :

https://github.com/jandhiesto/Tubes1_rudal-andriana/releases/tag/v.1.0

Link Bot Starter Pack:

https://github.com/Ariel-HS/tubes1-if2211-starter-pack/releases/tag/v1.0

DAFTAR PUSTAKA

Rinaldi Munir. Algoritma Greedy (Bagian 1). Diakses pada 10 Maret 2025 dari https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2024-2025/04-Algoritma-Greedy-(2025)-Bag1.pdf

Rinaldi Munir. Algoritma Greedy (Bagian 2). Diakses pada 10 Maret 2025 dari https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2024-2025/05-Algoritma-Greedy-(2025)-Bag2.pdf

Rinaldi Munir. Algoritma Greedy (Bagian 3). Diakses pada 10 Maret 2025 dari https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2024-2025/06-Algoritma-Greedy-(2025)-Bag3.pdf