King Abdulaziz University

Faculty of Engineering

Section: HA

Date 4/5/2025

Department of Industrial Engineering

IE322 – Computer Applications

Diamond and Gold Minerals Detector in Minecraft

No	Name	ID
1	Mohammad Alghamdi	2235778
2	Hosam Katoa	2338868
3	Abdulmajeed Almaswaie	2338913

4	Mohammed Balkhair	2339098
5	Muhannad Alsahafi	2340340

Instructor: Muhammad Atif Shahzad

Contents

1. Project Overview

This project focuses on the creation and implementation of an automated bot, named Steve, within the Minecraft game environment. The primary function of Steve is to autonomously mine valuable minerals, specifically gold and diamond ores. This bot enhances gameplay efficiency and showcases the potential for automation within a virtual sandbox environment.

2. Objectives

The primary objective of this project was to develop a bot capable of navigating underground environments to autonomously mine gold and diamond ores. The bot was designed to identify these specific minerals and extract them efficiently. Additionally, it aimed to optimize its movement paths to avoid unnecessary actions and reduce resource consumption. A further goal was to implement basic inventory management, ensuring that mined items were collected and stored properly. Lastly, the bot was built with safety protocols to avoid hazards such as lava, deep falls, and other environmental dangers that could disrupt its operation.

3. Tools and Technologies Used

This project was developed using Minecraft Java Edition, which allows for extensive customization through external libraries and scripts. The bot was programmed in JavaScript, utilizing the Mineflayer library along with Pathfinder and GoalBlock modules to enable navigation, block detection, and automated mining functions. To manage and run the bot, we used Node.js, a free and open-source JavaScript runtime environment that allowed us to set up a local server for executing our tools and scripts.

The coding process was carried out using the computer's built-in Notepad application for writing and editing scripts. Once the code was prepared, we used the Command Prompt window to run the Node.js server and launch the bot within the Minecraft environment. This simple but effective setup allowed for quick development and testing of the bot's features.

4. Bot Functionality

Steve the Bot was engineered to perform a variety of mining-related tasks autonomously. The bot was capable of scanning its surrounding environment to detect the presence of gold and diamond ores. Once these ores were located, it calculated an efficient and safe path to reach them. Upon arrival, the bot executed the appropriate mining actions to extract the ore, utilizing the correct tools for each block type. It then collected and stored the resources in its inventory. The bot also included logic to detect and avoid environmental hazards such as lava or unstable cave structures, ensuring continued operation without player intervention.

5. Implementation Details

The bot's operation began with an initialization phase, where it spawned at a specified location, typically near a mine or a resource-rich area. It would then begin scanning its surroundings in three dimensions to identify valuable ores. Once an ore was detected, the bot computed the shortest and safest route to reach it using pathfinding algorithms. When the target was reached, it would select the appropriate tool and begin mining. If any obstacles or hazards were encountered, the bot had built-in logic to reassess the situation and reroute if necessary. This cycle of scanning, pathfinding, and mining continued in a loop until the bot’s inventory was full or it was manually deactivated.

6. Challenges Faced

Several challenges were encountered during the development of Steve the Bot. One major challenge was accurately detecting ores in complex cave systems, especially when multiple types of blocks surrounded the target. Another significant issue involved ensuring the bot could avoid environmental hazards, such as sudden lava flows or dangerous drops. Additionally, optimizing the bot's pathfinding to reduce unnecessary movements required careful tuning of the algorithms used. Finally, balancing the bot’s performance and responsiveness in a real-time Minecraft environment proved to be a complex but rewarding task.

7. Results and Outcomes

The project successfully resulted in the creation of a functional mining bot that was able to autonomously locate and mine gold and diamond ores within the Minecraft world. Steve demonstrated considerable efficiency improvements compared to manual mining, significantly reducing the time and effort required to gather resources. The project highlighted the practical applications of automation and AI behavior in virtual environments. It also laid the groundwork for future expansions and feature enhancements.

8. Future Improvements

There are several areas where Steve the Bot can be improved in future versions. One potential enhancement is the implementation of more advanced scanning methods, such as simulated X-ray vision or machine learning-based block identification. Additionally, features like automatic return to base, resource sorting, and depositing items into chests could increase utility. Improving environmental interaction, such as using torches to light dark areas or placing water to neutralize lava, would further enhance safety. Finally, a graphical user interface (GUI) for easier control and monitoring of the bot could make the project more user-friendly.

9. Conclusion

Steve the mining bot represents a significant step forward in blending automation with interactive gameplay in Minecraft. By independently mining valuable resources such as gold and diamond, the bot adds efficiency and intelligence to the game experience. This project serves as an example of how programming skills can be applied creatively in virtual environments, combining entertainment with learning and innovation. With further improvements, Steve has the potential to become a comprehensive mining assistant within the Minecraft universe.

References

Baritone (n.d.) *Baritone: The pathfinding bot used in Minecraft*. Available at: https://github.com/cabaletta/baritone (Accessed: 1 May 2025).

FabricMC (n.d.) *Fabric Modding Toolchain*. Available at: https://fabricmc.net/ (Accessed: 1 May 2025).

Mineflayer (n.d.) *Mineflayer – Create Minecraft bots with JavaScript*. Available at: https://github.com/PrismarineJS/mineflayer (Accessed: 1 May 2025).

Minecraft (n.d.) *Minecraft Java Edition*. Mojang Studios. Available at: https://www.minecraft.net/en-us (Accessed: 1 May 2025).

Node.js (n.d.) *Node.js®. JavaScript runtime built on Chrome's V8 JavaScript engine*. Available at: https://nodejs.org/ (Accessed: 1 May 2025).

Pathfinder (n.d.) *mineflayer-pathfinder plugin*. Available at: https://github.com/PrismarineJS/mineflayer-pathfinder (Accessed: 1 May 2025).

Visual Studio Code (n.d.) *Code editing. Redefined – Visual Studio Code*. Microsoft. Available at: https://code.visualstudio.com/ (Accessed: 1 May 2025).

Appendices

Bot coding:

```
“const mineflayer = require('mineflayer')
const { pathfinder, Movements, goals } = require('mineflayer-pathfinder')
const { GoalBlock } = goals

const bot = mineflayer.createBot({
  host: 'localhost',
  port: 25565,
  username: 'SteveBot',
  version: '1.20.4'
})

bot.loadPlugin(pathfinder)

bot.once('spawn', () => {
  const mcData = require('minecraft-data')(bot.version)
  const defaultMove = new Movements(bot, mcData)
  bot.pathfinder.setMovements(defaultMove)

  findAndMine()
})

function findAndMine() {
  const target = bot.findBlock({
    matching: block => block && (
      block.name === 'diamond_ore' || block.name === 'gold_ore'
```

```
  ),
  maxDistance: 32
})

if (!target) {
  console.log('❌ No diamond or gold ore nearby. Scanning again...')
  setTimeout(findAndMine, 2000)
  return
}

console.log(` 🪓 Found ${target.name}! Going to mine... `)
const { x, y, z } = target.position
const goal = new GoalBlock(x, y, z)
bot.pathfinder.setGoal(goal)

bot.once('goal_reached', () => {
  const blockAtTarget = bot.blockAt(target.position)

  if (!blockAtTarget || (blockAtTarget.name !== 'diamond_ore' && blockAtTarget.name
!== 'gold_ore')) {
    console.log('🤷 Block already gone. Scanning again...')
    return setTimeout(findAndMine, 1000)
  }

  bot.dig(blockAtTarget, err => {
    if (err) {
      console.log('❌ Error digging:', err.message)
    } else {
```

```
      console.log(` ✅ Mined ${blockAtTarget.name}!`)
    }
    setTimeout(findAndMine, 1000)
  })
 })
}

bot.on('error', err => console.log('Bot error:', err))
bot.on('end', () => console.log('Bot disconnected.'))”.
```