ANNEX

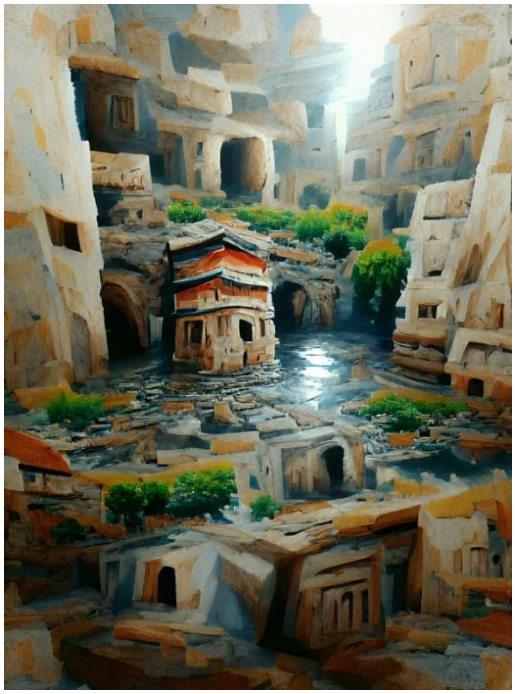
Autor: Guillem Cabré Farré
Tutors: • Joan Álamo Bretcha
• Montserrat Beltran i Berengué
Curs: 2n Batxillerat 2021-22
INS Arquitecte Manuel Raspall
Àmbit científicotecnològic

29 d'octubre de 2021

ÍNDEX

ANNEX 1

Aquestes il·lustracions han estat creades mitjançant la intel·ligència artificial VQGAN+CLIP, s'ha parlat d'aquesta anteriorment, a l'apartat: Quines aplicacions té? El peu d'imatge és el text que se li va introduir a la IA.



Il·lustració 1: Ancient city oil painting unreal engine hyperrealistic.

Il·lustració 2: Foggy and mossy postapocalyptic village unreal engine hyperrealistic.

Il·lustració 3: Nature futuristic city.

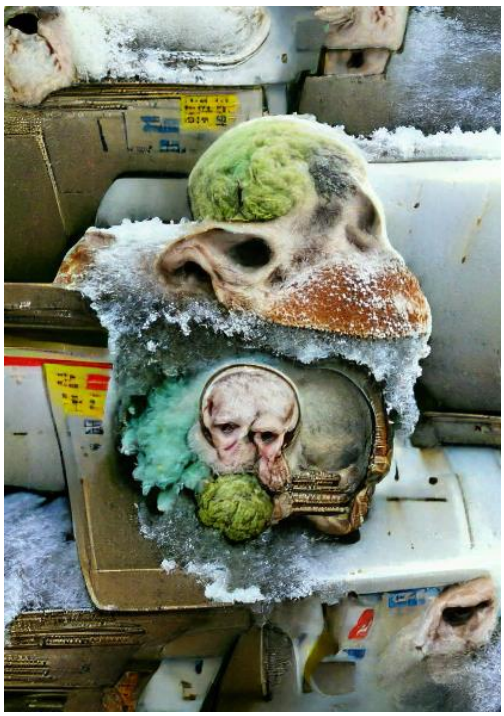
Il·lustració 4: Futuristic city raining.

Il·lustració 5: Explosion in village unreal engine hyperrealistic.

Il·lustració 6: Sunset mystic city with sea.

Il·lustració 7: Snowy mountain scenary lake sunny unreal engine hyperrealistic.

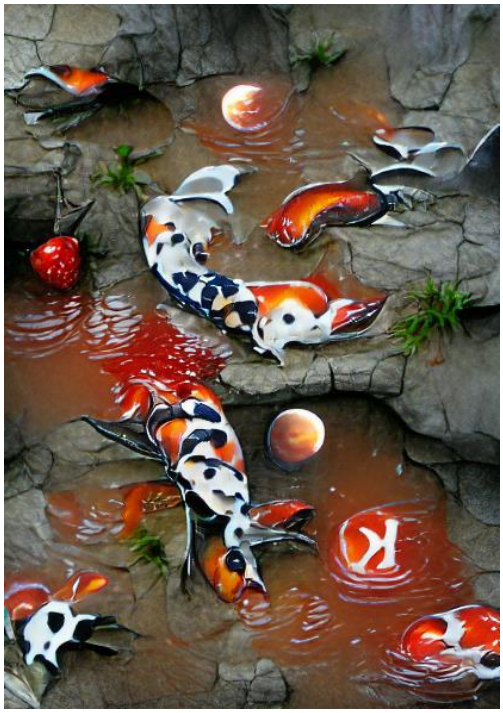
Il·lustració 8: Dark sinister village unreal engine hyperrealistic.



Il·lustració 9: Frozen moldy skull.

Il·lustració 10: Dark fog scenary.

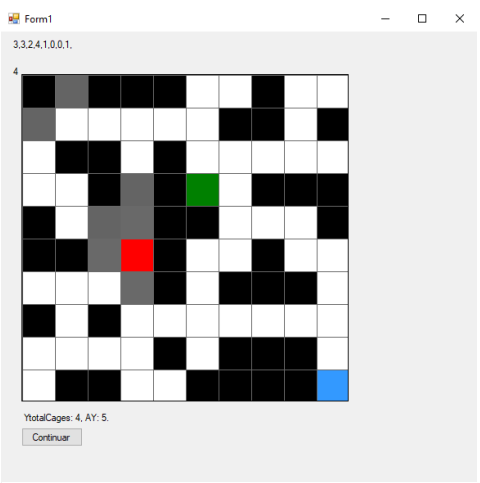
Il·lustració 11: Shrines in the distance.



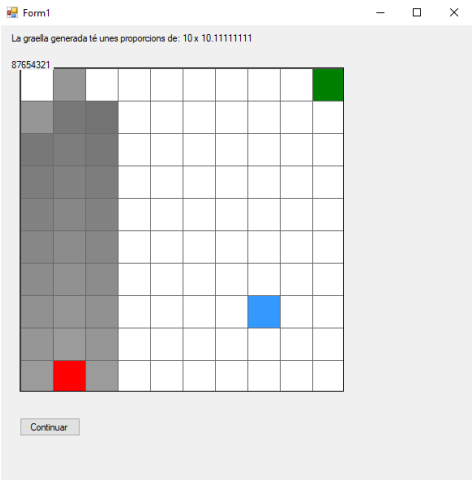
Il·lustració 12: Koi fish dreamland.

ANNEX 2

Durant la creació del programa vaig topar-me amb una gran quantitat de "bugs", errors que apareixen quan no es contemplen totes les possibilitats o quan no es compleix el que el programador pretén. De tots els "bugs" que em van aparèixer en vaig escollir alguns per afegir-los aquí, ja que em va semblar interessant exposar les dificultats que he tingut.



Il·lustració 13: Primer bug.



Il·lustració 14: Segon bug.

A la il·lustració 5 es pot veure com la IA avança dues iteracions d'una manera errònia, ja que no pinta les cel·les de baix. A més, pinta dues cel·les a dalt de tot a l'esquerra, que haurien d'estar pintades d'un altre color, ja que no correspon amb la iteració que li pertocaria. El mateix passa a la següent il·lustració, on podem veure que hi ha dues cel·les amb un color erroni a dalt a l'esquerra.

Aquest error apareixia degut a una falla en el tros de codi que separa les coordenades X i Y de les cel·les des de les quals es fa l'avenç a la següent iteració.

He passat per molts més "bugs", però la majoria provoquen que es penji el programa i no són gaire interessants. Solien passar quan una variable es sortia dels seus límits dins d'una condició.

ANNEX 3

A continuació he penjat el codi sencer del programa. Compta amb un total de 530 línies i és el responsable de fer funcionar l'aplicació que s'ha vist anteriorment. A més a més, he adjuntat la carpeta del programa en el sisè enllaç d'*altres enllaços*. Per executar-lo un cop descarregada la carpeta s'ha de fer doble clic en l'arxiu: EXECUTAR PROGRAMA. Per trobar el codi, s'ha d'obrir amb el programa Visual Studio o un altre compilador de text, o sinó amb el bloc de notes l'arxiu dins de la carpeta "AppPathFinding" anomenat: Form1.cs. Dins de la carpeta "bin" i després dins de "Debug", hi ha la capreta "GrillSelection", on es poden veure i modificar les diferents graelles guardades dins del programa. Les anomenades: SetGrill, són les predeterminades i CreatedGrill l'última creada per l'usuari.

El codi està organitzat amb separadors i anotacions per facilitar l'enteniment d'aquest. Les anotacions estan en cursiva darrera de dues barres (//(...)). I els separadors tenen molts guionets baixos seguits del títol en majúscules.

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Diagnostics;
using System.Drawing;
using System.IO;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace AppPathFinding
{
  public partial class Form1 : Form
  {
    public int Xcages; //Amplària de la graella creació
    public int Ycages; //Altura de la graella creació
    public int CreateGrillState; //Indica l'estat de creació de la graella
    public bool Asetted = false; //true si ja s'ha posat la A a la creació de la graella
    public bool Bsetted = false; //true si ja s'ha posat la B a la creació de la graella
    //Dimensions de la graella
    public int YtotalCages; //Altura graella
    public int XtotalCages; //Amplària graella
    //Coordenades punt A, no hi ha B perquè la IA l'haurà de trobar, no se li donarà la informació
    public int AX; //Coordenada lateral
    public int AY; //Coordenada vertical
```

```
    int ColorJumps = 5; //MODIFICABLE! Inserir cada quants vallors sobre 255 es vol que es facin els salts

    public string filePath;

    public Form1()
    {
        InitializeComponent();
    }

    private void btn_startprogram_Click(object sender, EventArgs e)
    {
        btn_startprogram.Visible = false;
        lbl_Speaker1.Text = "Escull una de les opcions que hi ha a sota:";
        btn_CreateGrill.Visible = true;
        btn_ChooseGrill.Visible = true;
        btn_RandomGrill.Visible = true;
    }

    private void btn_CreateGrill_Click(object sender, EventArgs e)
    {
        lbl_Speaker1.Text = "Has escollit crear una graella.";
        lbl_Speaker2.Text = $"Insereix el nombre de cel·les en l'eix de les x:";
        lbl_Speaker3.Text = $"Insereix el nombre de cel·les en l'eix de les y:";
        num_Xcages.Visible = true;
        num_Ycages.Visible = true;
        btn_CreateGrill.Visible = false;
        btn_ChooseGrill.Visible = false;
        btn_RandomGrill.Visible = false;
    }

    private void num_Xcages_ValueChanged(object sender, EventArgs e)
    {
        if (num_Xcages.Value + num_Ycages.Value == 2) { btn_ProceedCreateGrill.Visible = false; } //Es vigila que no es faci una graella 1*1
        else { btn_ProceedCreateGrill.Visible = true; }
    }

    private void num_Ycages_ValueChanged(object sender, EventArgs e)
    {
        Ycages = Convert.ToInt32(Math.Round(num_Ycages.Value, 0));
        if (num_Xcages.Value + num_Ycages.Value == 2) { btn_ProceedCreateGrill.Visible = false; }
        else { btn_ProceedCreateGrill.Visible = true; }
    }

    private void btn_ProceedCreateGrill_Click(object sender, EventArgs e)
    {
        Xcages = Convert.ToInt32(Math.Round(num_Xcages.Value, 0));
        Ycages = Convert.ToInt32(Math.Round(num_Ycages.Value, 0));
        Grill.Visible = true;
        lbl_Speaker1.Text = $"La teva graella és així: x= {Xcages},   y= {Ycages}";
        lbl_Speaker2.Text = "";
        lbl_Speaker3.Text = "";
        btn_ProceedCreateGrill.Visible = false;
        num_Xcages.Visible = false;
        num_Ycages.Visible = false;
```

```
        for (int count = 0; count < Xcages; count++)
        {
            Grill.Columns.Add("", "");
            Grill.Columns[count].Width = (400 / (Xcages));
        }
        for (int count = 0; count < Ycages; count++)
        {
            Grill.Rows.Add("", "");
            Grill.Rows[count].Height = (400 / Ycages);
        }
        lbl_BelowGrill.Text = "T'agrada el resultat? o el vols repetir?";
        btn_GrillProceed.Visible = true;
        btn_GrillRepeat.Visible = true;
    }

    private void btn_GrillRepeat_Click(object sender, EventArgs e)
    {
        Grill.Visible = false;
        btn_GrillRepeat.Visible = false;
        btn_GrillProceed.Visible = false;
        lbl_BelowGrill.Text = "";
        lbl_Speaker1.Text = "Has escollit crear una graella.";
        lbl_Speaker2.Text = "Insereix el nombre de cel·les en l'eix de les x:";
        lbl_Speaker3.Text = "Insereix el nombre de cel·les en l'eix de les y:";
        num_Xcages.Visible = true;
        num_Ycages.Visible = true;
        btn_CreateGrill.Visible = false;
        btn_ChooseGrill.Visible = false;
        btn_RandomGrill.Visible = false;
        btn_ProceedCreateGrill.Visible = true;
    }

    private void btn_GrillProceed_Click(object sender, EventArgs e)
    {

        if (CreateGrillState == 0)
        {
            lbl_Speaker1.Text = "Ja pots començar a omplir la graella. Clica en la cel·la en la que vols operar i presiona els botons que
t'interessin.";
            lbl_BelowGrill.Text = "";
            btn_InputWallVoid.Visible = true;
            btn_InputAPoint.Visible = true;
            btn_InputBPoint.Visible = true;
            btn_GrillProceed.Visible = false;
            btn_GrillRepeat.Visible = false;
            for (int Ycount = 0; Ycount < Ycages; Ycount++)
            {
                for (int Xcount = 0; Xcount < Xcages; Xcount++)
                {
                    Grill.Rows[Ycount].Cells[Xcount].Style.BackColor = Color.White;
                }
            }
            CreateGrillState = 1;
            return;
        }
        if (CreateGrillState == 1)
        {
```

```
            btn_InputWallVoid.Visible = false;
            btn_InputAPoint.Visible = false;
            btn_InputBPoint.Visible = false;
            lbl_BelowGrill.Text = "Ja has creat la teva graella.";
            lbl_OnlyOneA.Visible = false;
            lbl_OnlyOneB.Visible = false;
            lbl_Speaker1.Text = "";
            btn_GrillProceed.Visible = false;
            filePath = Path.GetDirectoryName(Process.GetCurrentProcess().MainModule.FileName) + @"\GrillSelection\CreatedGrill.txt";
//Aconsegueix el directori relatiu .txt
            List<string> lines = new List<string>();
            for (int Ycount = 0; Ycount < Ycages; Ycount++)
            {
                string lineFile = null;
                for (int Xcount = 0; Xcount < Xcages; Xcount++)
                {
                    if (Grill.Rows[Ycount].Cells[Xcount].Style.BackColor == Color.Black) { lineFile = $"{lineFile}X "; }      //Aquestes quatre
linies detecten el color de les cel·les per convertir les linies en string i finalment posar-les en el .txt
                    else if (Grill.Rows[Ycount].Cells[Xcount].Style.BackColor == Color.White) { lineFile = $"{lineFile}- "; }
                    else if (Grill.Rows[Ycount].Cells[Xcount].Style.BackColor == Color.Red) { lineFile = $"{lineFile}A "; }
                    else if (Grill.Rows[Ycount].Cells[Xcount].Style.BackColor == Color.Green) { lineFile = $"{lineFile}B "; }
                }
                lines.Add(lineFile);
                File.WriteAllLines(filePath, lines);
            }
            ShowSetGrill();
        }
        if (CreateGrillState == 2)
        {
            lbl_SetGrillOptions.Visible = false;
            btn_SetGrill1.Visible = false;
            btn_SetGrill2.Visible = false;
            btn_SetGrill3.Visible = false;
            InicializatePathfinding();
        }
    }

    private void btn_InputWallVoid_Click(object sender, EventArgs e)
    {
        lbl_OnlyOneA.Visible = false;
        lbl_OnlyOneB.Visible = false;
        if ((Grill.CurrentCell.Style.BackColor == Color.Red))
        {
            Grill.CurrentCell.Style.BackColor = Color.Black;
            Asetted = false;
            btn_GrillProceed.Visible = false;
        }
        if ((Grill.CurrentCell.Style.BackColor == Color.Green))
        {
            Grill.CurrentCell.Style.BackColor = Color.Black;
            Bsetted = false;
            btn_GrillProceed.Visible = false;
        }
        if (Grill.CurrentCell.Style.BackColor != Color.Black) { Grill.CurrentCell.Style.BackColor = Color.Black; } //Detecta si la cel·la
seleccionada és pared, si ho és ho canvia a buit i si no, a pared
        else { Grill.CurrentCell.Style.BackColor = Color.White; }
    }
```

```
    private void btn_InputAPoint_Click(object sender, EventArgs e) //Insereix el punt inicial de PathFindeing
    {
        lbl_OnlyOneA.Visible = false;
        lbl_OnlyOneB.Visible = false;
        if (Asetted == false)
        {
            if (Grill.CurrentCell.Style.BackColor == Color.Green) { Bsetted = false; } //Detecta si està a sobre del punt B per canviar
Bsetted a false
            Asetted = true;
            Grill.CurrentCell.Style.BackColor = Color.Red;
            if (Asetted == true && Bsetted == true)
            {
                btn_GrillProceed.Visible = true;
            }
        }
        else { lbl_OnlyOneA.Visible = true; }
    }

    private void btn_InputBPoint_Click(object sender, EventArgs e) //Insereix punt final de PathFindeing
    {
        lbl_OnlyOneA.Visible = false;
        lbl_OnlyOneB.Visible = false;
        if (Bsetted == false)
        {
            if (Grill.CurrentCell.Style.BackColor == Color.Red) { Asetted = false; }
            Bsetted = true;
            Grill.CurrentCell.Style.BackColor = Color.Green;
            if (Asetted == true && Bsetted == true)
            {
                btn_GrillProceed.Visible = true;
            }
        }
        else { lbl_OnlyOneB.Visible = true; }
    }

    //_____________________________________________________________________________ ALTRE STARTING
OPTION

    private void btn_ChooseGrill_Click_1(object sender, EventArgs e)
    {
        btn_startprogram.Visible = false;
        btn_CreateGrill.Visible = false;
        btn_ChooseGrill.Visible = false;
        btn_RandomGrill.Visible = false;
        lbl_Speaker1.Text = "";
        Grill.Visible = true;
        lbl_SetGrillOptions.Visible = true;
        btn_SetGrill1.Visible = true;
        btn_SetGrill2.Visible = true;
        btn_SetGrill3.Visible = true;
    }
    private void btn_SetGrill1_Click(object sender, EventArgs e)
    {
        filePath = Path.GetDirectoryName(Process.GetCurrentProcess().MainModule.FileName) + @"\GrillSelection\SetGrill1.txt";
//Aconsegueix el directori relatiu .txt
        ShowSetGrill();
```

```
    }
    private void btn_SetGrill2_Click(object sender, EventArgs e)
    {
        filePath = Path.GetDirectoryName(Process.GetCurrentProcess().MainModule.FileName) + @"\GrillSelection\SetGrill2.txt";
//Aconsegueix el directori relatiu .txt
        ShowSetGrill();
    }
    private void btn_SetGrill3_Click(object sender, EventArgs e)
    {
        filePath = Path.GetDirectoryName(Process.GetCurrentProcess().MainModule.FileName) + @"\GrillSelection\SetGrill3.txt";
//Aconsegueix el directori relatiu .txt
        ShowSetGrill();
    }

    //_____________________________________________________________________ ALTRE STARTING
OPTION

    private void btn_RandomGrill_Click(object sender, EventArgs e)
    {
        btn_startprogram.Visible = false;
        btn_CreateGrill.Visible = false;
        btn_ChooseGrill.Visible = false;
        btn_RandomGrill.Visible = false;
        Random random = new Random();
        int randomnumber = random.Next(4); //Quatre opcions: 0, 1, 2, 3. 3 = última graella creada per l'usuari.
        switch (randomnumber)
        {
            case 0:
                filePath = Path.GetDirectoryName(Process.GetCurrentProcess().MainModule.FileName) + @"\GrillSelection\SetGrill1.txt";
                break;
            case 1:
                filePath = Path.GetDirectoryName(Process.GetCurrentProcess().MainModule.FileName) + @"\GrillSelection\SetGrill2.txt";
                break;
            case 2:
                filePath = Path.GetDirectoryName(Process.GetCurrentProcess().MainModule.FileName) + @"\GrillSelection\SetGrill3.txt";
                break;
            case 3:
                filePath = Path.GetDirectoryName(Process.GetCurrentProcess().MainModule.FileName) +
@"\GrillSelection\CreatedGrill.txt"; //Última creada
                break;
        }
        Grill.Visible = true;
        ShowSetGrill();
    }

    //_____________________________________________________________________ SHOW SET GRILL

    private void ShowSetGrill()
    {
        lbl_BelowGrill.Text = "";
        Grill.Rows.Clear();
        Grill.Columns.Clear(); //Elimina qualsevol tipus de graella possible
        List<string> lines = new List<string>();
        lines = File.ReadAllLines(filePath).ToList();
        YtotalCages = 0; //Es guarda en aquesta variable el número de cel·les que ha de tenir el gràfic
        using (StreamReader r = new StreamReader(filePath)) //Conta totes les linies del .txt
        {
```

```
            while (r.ReadLine() != null) { YtotalCages++; } //Quan detecta una linia nova suma 1 a "YtotalCages"
        }
        XtotalCages = 0;
        foreach (String line in lines) { XtotalCages = line.Length / 2; } //Dividim entre dos per tenir en compte els espais entre cada lletra
        for (int Xcount = 0; Xcount < XtotalCages; Xcount++) //Crea les columnes
        {
            Grill.Columns.Add("", "");
            Grill.Columns[Xcount].Width = (400 / XtotalCages);
        }
        for (int Ycount = 0; Ycount < YtotalCages; Ycount++) //Crea les files
        {
            Grill.Rows.Add("", "");
            Grill.Rows[Ycount].Height = (400 / YtotalCages);
        }
        lbl_Speaker1.Text = $"La graella generada té unes proporcions de: {YtotalCages} x {XtotalCages}.";
        int YcolorCount = 0;
        foreach (String line in lines) //Loop que es repeteix per cada linia
        {
            string[] cage = line.Split(' '); //Talla el string cada cop que hi ha un "espai"
            for (int XcolorCount = 0; XcolorCount < XtotalCages; XcolorCount++)
            {
                switch (cage[XcolorCount]) //TRANSCRIPTOR d'arxius .txt
                {
                    case "X":
                        Grill.Rows[YcolorCount].Cells[XcolorCount].Style.BackColor = Color.Black;
                        break;
                    case "A":
                        Grill.Rows[YcolorCount].Cells[XcolorCount].Style.BackColor = Color.Red;
                        AX = XcolorCount;   //Defineix en AX i AY les coordenades del punt A
                        AY = YcolorCount;
                        break;
                    case "B":
                        Grill.Rows[YcolorCount].Cells[XcolorCount].Style.BackColor = Color.Green;
                        break;
                    case "-":
                        Grill.Rows[YcolorCount].Cells[XcolorCount].Style.BackColor = Color.White;
                        break;
                }
                Grill.Rows[YcolorCount].Cells[XcolorCount].Selected = false;
            }
            YcolorCount++;
        }
        btn_GrillProceed.Visible = true;
        CreateGrillState = 2;
    }



    //_____________________________________ INICI PATHFINDING _____________________________________

    private void InicializatePathfinding()
    {
        //_____________________PRIMERA PART IA_____________________

        btn_GrillProceed.Visible = false;
```

```
        lbl_Speaker1.Text = "";
        btn_ProceedCreateGrill.Visible = false;
        int ColorTimes = 50; //Intensitat del color del primer pas
        bool PathfindingForward = true; //Responsable d'apagar la la part del PathFinding
        int NumberOfNewChecks = 1; //Només 1 check, que és el punt A
        int i = 0; //Numero d'iteracions de la IA
        string MemoryXY = ""; //Aquí s'emmagatzemen les coordenades dels avenços.
        int BXcoords = 0; // |
        int BYcoords = 0; // | Coordenades B

        while (PathfindingForward == true) //Es repeteix fins que es troba B
        {
            int[] XcoordsProcess = new int[NumberOfNewChecks]; // |
            int[] YcoordsProcess = new int[NumberOfNewChecks];// |Serveixen per guardar les coordenades dels nous checks. Un per la coordenada X, l'altre per la Y. Hi ha tants ints com NumOfNewChecks
            if (i == 0) //Si és la iteració 0 (primera), assigna les coords a cheks del punt A
            {
                XcoordsProcess[0] = AX;
                YcoordsProcess[0] = AY;
            }
            if (i != 0) //Si no és la iteració 0 (primera)
            {
                string[] Memory = MemoryXY.Split(','); //Talla el string cada cop que hi ha un "coma"
                for (int repetitions = 0; repetitions < NumberOfNewChecks; repetitions++) //Passa string (Memory[]) que conté x1,y1,x2,y2.... a x variables i y variables
                {
                    XcoordsProcess[repetitions] = Int16.Parse(Memory[repetitions * 2]);
                    YcoordsProcess[repetitions] = Int16.Parse(Memory[repetitions * 2 + 1]); //Com que principalment està: x1,y1,x2,y2... se li suma 1 perquè agafi les impars
                }
            }
            MemoryXY = ""; //Torna la memòria al valor inicial

            int SumOfChecks = 0; //Conta quants checks s'han de fer a la següent "i"
            int ChecksCount = 0;

            while (ChecksCount < NumberOfNewChecks)
            {
                if (YcoordsProcess[ChecksCount] > 0) //Detecta que hi ha adalt, si hi ha graella
                {
                    if (Grill.Rows[YcoordsProcess[ChecksCount] - 1].Cells[XcoordsProcess[ChecksCount]].Style.BackColor == Color.White) //Si es "white" pinta amb escala de colors
                    {
                        Grill.Rows[YcoordsProcess[ChecksCount] - 1].Cells[XcoordsProcess[ChecksCount]].Style.BackColor = Color.FromArgb(255, 255, 255 - ColorTimes);
                        MemoryXY = $"{MemoryXY}{XcoordsProcess[ChecksCount]},{YcoordsProcess[ChecksCount] - 1},"; //Afageix a la memòria la cel·la a la que s'ha avançat
                        SumOfChecks++;
                    }
                    if (Grill.Rows[YcoordsProcess[ChecksCount] - 1].Cells[XcoordsProcess[ChecksCount]].Style.BackColor == Color.Green) //Si es "green" ves a la segona part del PathFinding
                    {
                        PathfindingForward = false;
                        BYcoords = YcoordsProcess[ChecksCount] - 1;
                        BXcoords = XcoordsProcess[ChecksCount];
                    }
                }
```

```
            if (XcoordsProcess[ChecksCount] < (XtotalCages - 1)) //Detecta que hi ha a la dreta, si hi ha graella
            {
              if (Grill.Rows[YcoordsProcess[ChecksCount]].Cells[XcoordsProcess[ChecksCount] + 1].Style.BackColor == Color.White)
              {
                Grill.Rows[YcoordsProcess[ChecksCount]].Cells[XcoordsProcess[ChecksCount] + 1].Style.BackColor =
Color.FromArgb(255, 255 , 255 - ColorTimes);
                MemoryXY = $"{MemoryXY}{XcoordsProcess[ChecksCount] + 1},{YcoordsProcess[ChecksCount]},"; //Afageix a la
memòria la cel·la a la que s'ha avançat
                SumOfChecks++;
              }
              if (Grill.Rows[YcoordsProcess[ChecksCount]].Cells[XcoordsProcess[ChecksCount] + 1].Style.BackColor ==
Color.Green) //Si es "green" ves a la segona part del PathFinding
              {
                PathfindingForward = false;
                BYcoords = YcoordsProcess[ChecksCount];
                BXcoords = XcoordsProcess[ChecksCount] + 1;
              }
            }

            if (YcoordsProcess[ChecksCount] < (YtotalCages - 1)) //Detecta que hi ha abaix, si hi ha graella
            {
              if (Grill.Rows[YcoordsProcess[ChecksCount] + 1].Cells[XcoordsProcess[ChecksCount]].Style.BackColor == Color.White)
              {
                Grill.Rows[YcoordsProcess[ChecksCount] + 1].Cells[XcoordsProcess[ChecksCount]].Style.BackColor =
Color.FromArgb(255 , 255 , 255 - ColorTimes);
                MemoryXY = $"{MemoryXY}{XcoordsProcess[ChecksCount]},{YcoordsProcess[ChecksCount] + 1},"; //Afageix a la
memòria la cel·la a la que s'ha avançat
                SumOfChecks++;
              }
              if (Grill.Rows[YcoordsProcess[ChecksCount] + 1].Cells[XcoordsProcess[ChecksCount]].Style.BackColor ==
Color.Green) //Si es "green" ves a la segona part del PathFinding
              {
                PathfindingForward = false;
                BYcoords = YcoordsProcess[ChecksCount] + 1;
                BXcoords = XcoordsProcess[ChecksCount];
              }
            }

            if (XcoordsProcess[ChecksCount] > 0) //Detecta que hi ha a l'esquerra, si hi ha graella
            {
              if (Grill.Rows[YcoordsProcess[ChecksCount]].Cells[XcoordsProcess[ChecksCount] - 1].Style.BackColor == Color.White)
              {
                Grill.Rows[YcoordsProcess[ChecksCount]].Cells[XcoordsProcess[ChecksCount] - 1].Style.BackColor =
Color.FromArgb(255 , 255 , 255 - ColorTimes);
                MemoryXY = $"{MemoryXY}{XcoordsProcess[ChecksCount] - 1},{YcoordsProcess[ChecksCount]},"; //Afageix a la
memòria la cel·la a la que s'ha avançat
                SumOfChecks++;
              }
              if (Grill.Rows[YcoordsProcess[ChecksCount]].Cells[XcoordsProcess[ChecksCount] - 1].Style.BackColor == Color.Green)
//Si es "green" ves a la segona part del PathFinding
              {
                PathfindingForward = false;
                BYcoords = YcoordsProcess[ChecksCount];
                BXcoords = XcoordsProcess[ChecksCount] - 1;
              }
            }
```

```
            ChecksCount++;
        }
        if (SumOfChecks == 0) //Detecta si és impossible arribar a l'objectiu
        {
            PathfindingForward = false;
            lbl_Speaker1.Text = "La graella escollida no té una connexió directa entre el punt A i el punt B. Prova una altra vegada.";
            btn_RepetirApp.Visible = true;
        }

        if (PathfindingForward == true) { ColorTimes += ColorJumps; } //Si no s'ha acabat, afegeix 5 al color
        NumberOfNewChecks = SumOfChecks;
        i++;
    }
    lbl_BelowGrill.Text = $"S'han fet un total de {i} iteracions.";

    //_____________________SEGONA PART IA_____________________

    int TimesSecond = 0;
    int RelativeX = BXcoords; // |
    int RelativeY = BYcoords;// |Posicions relatives de l'avenç, assignades a les coordenades del punt B

    while (TimesSecond < i)
    {
        if (RelativeY > 0) //Detecta que hi ha adalt, si hi ha graella
        {
            if (Grill.Rows[RelativeY - 1].Cells[RelativeX].Style.BackColor == Color.FromArgb(255, 255, 255 + ColorJumps - ColorTimes + (ColorJumps * TimesSecond))) //Adalt
            {
                Grill.Rows[RelativeY - 1].Cells[RelativeX].Style.BackColor = Color.FromArgb(0 + (255/i) * TimesSecond, 255 - (255 / i) * TimesSecond, 0); //Pintar gradualment
                RelativeY += -1;
            }
        }

        if (RelativeX  < (XtotalCages - 1)) //Detecta que hi ha a la dreta, si hi ha graella
        {
            if (Grill.Rows[RelativeY].Cells[RelativeX + 1].Style.BackColor == Color.FromArgb(255, 255, 255 + ColorJumps - ColorTimes + (ColorJumps * TimesSecond))) //Dreta
            {
                Grill.Rows[RelativeY].Cells[RelativeX + 1].Style.BackColor = Color.FromArgb(0 + (255 / i) * TimesSecond, 255 - (255 / i) * TimesSecond, 0); //Pintar gradualment
                RelativeX++;
            }
        }

        if (RelativeY < (YtotalCages - 1)) //Detecta que hi ha adalt, si hi ha graella
        {
            if (Grill.Rows[RelativeY + 1].Cells[RelativeX].Style.BackColor == Color.FromArgb(255, 255, 255 + ColorJumps - ColorTimes + (ColorJumps * TimesSecond))) //Adalt
            {
                Grill.Rows[RelativeY + 1].Cells[RelativeX].Style.BackColor = Color.FromArgb(0 + (255 / i) * TimesSecond, 255 - (255 / i) * TimesSecond, 0); //Pintar gradualment
                RelativeY++;
            }
        }

        if (RelativeX > 0) //Detecta que hi ha a la esquerra, si hi ha graella
```

```
            {
                if (Grill.Rows[RelativeY].Cells[RelativeX - 1].Style.BackColor == Color.FromArgb(255, 255, 255 + ColorJumps - ColorTimes
+ (ColorJumps * TimesSecond))) //Esquerra
                {
                    Grill.Rows[RelativeY].Cells[RelativeX - 1].Style.BackColor = Color.FromArgb(0 + (255 / i) * TimesSecond, 255 - (255 / i)
* TimesSecond, 0); //Pintar gradualment
                    RelativeX += -1;
                }
            }
            TimesSecond++;
        }
        btn_RepetirApp.Visible = true;
    }

    private void btn_RepetirApp_Click(object sender, EventArgs e)
    {
        Grill.Visible = false;
        Grill.Rows.Clear();
        lbl_BelowGrill.Text = "";
        lbl_Speaker1.Text = "";
        CreateGrillState = 0;
        Asetted = false;
        Bsetted = false;
        btn_startprogram.Visible = true;
        btn_RepetirApp.Visible = false;
    }
  }
}
```