Daniel Kaestner, Olive Kirk, Basia Futyma, Thalea Schlender and Stijn Coppens

Crazy Putting!

Project Group 10

Introduction

Objectives:

Course Builder

Physics Engine

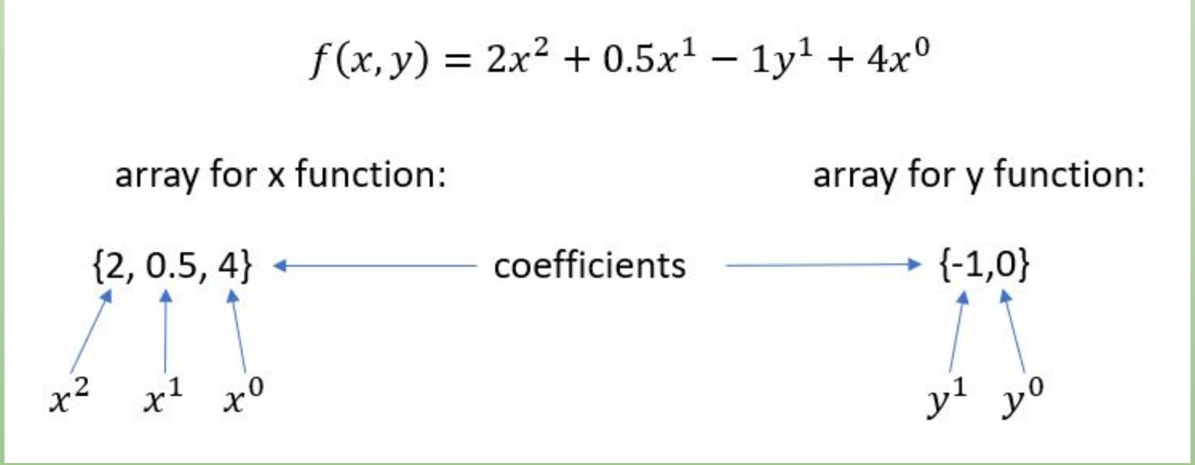
Artificial Intelligence

Cooperative Multiplayer

Experiments

Course Builder

Height function



$$f(x,y) = 2x^2 + 0.5x^1 - 1y^1 + 4x^0$$

array for x function:

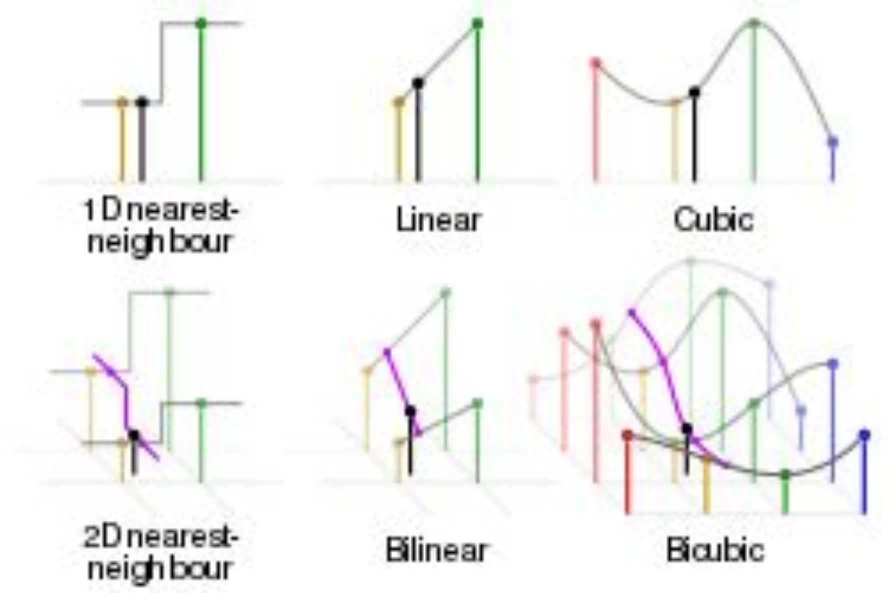
array for y function:

{2, 0.5, 4} ← coefficients → {-1,0}

x^2 x^1 x^0

y^1 y^0

Spline Interpolation



Bicubic Spline Interpolation

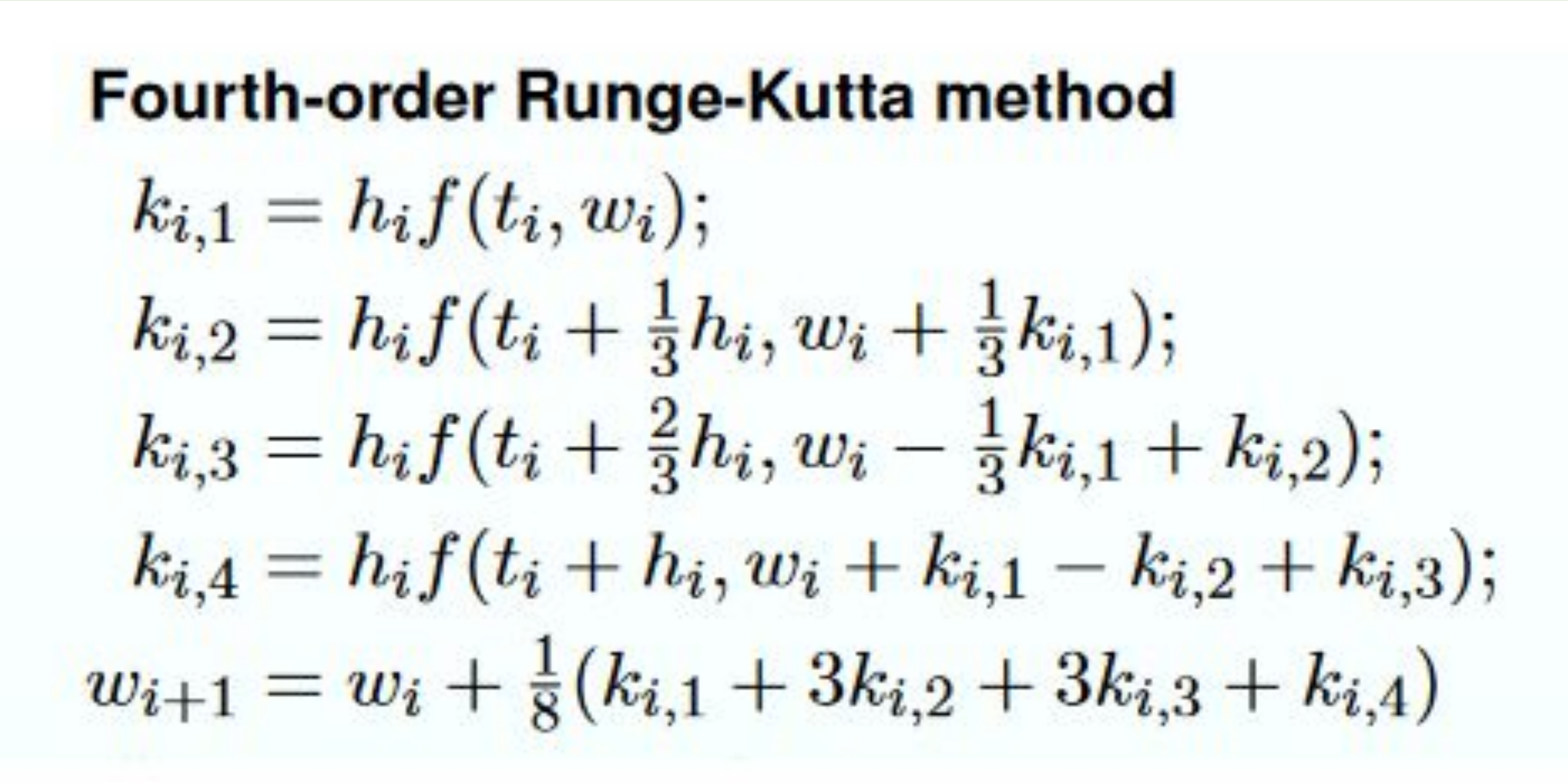
$$\begin{bmatrix} a_{00} & a_{01} & a_{02} & a_{03} \\ a_{10} & a_{11} & a_{12} & a_{13} \\ a_{20} & a_{21} & a_{22} & a_{23} \\ a_{30} & a_{31} & a_{32} & a_{33} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -3 & 3 & -2 & -1 \\ 2 & -2 & 1 & 1 \end{bmatrix} \begin{bmatrix} f(x_{00}, y_{00}) & f(x_{01}, y_{01}) & f_y(x_{00}, y_{00}) & f_y(x_{01}, y_{01}) \\ f(x_{10}, y_{10}) & f(x_{11}, y_{11}) & f_y(x_{10}, y_{10}) & f_y(x_{11}, y_{11}) \\ f_x(x_{00}, y_{00}) & f_x(x_{01}, y_{01}) & f_{xy}(x_{00}, y_{00}) & f_{xy}(x_{01}, y_{01}) \\ f_x(x_{10}, y_{10}) & f_x(x_{11}, y_{11}) & f_{xy}(x_{10}, y_{10}) & f_{xy}(x_{11}, y_{11}) \end{bmatrix} \begin{bmatrix} 1 & 0 & -3 & 2 \\ 0 & 0 & 3 & -2 \\ 0 & 1 & -2 & 1 \\ 0 & 0 & -1 & 1 \end{bmatrix}$$

$$p(x,y) = \begin{bmatrix} 1 & x & x^2 & x^3 \end{bmatrix} \begin{bmatrix} a_{00} & a_{01} & a_{02} & a_{03} \\ a_{10} & a_{11} & a_{12} & a_{13} \\ a_{20} & a_{21} & a_{22} & a_{23} \\ a_{30} & a_{31} & a_{32} & a_{33} \end{bmatrix} \begin{bmatrix} 1 \\ y \\ y^2 \\ y^3 \end{bmatrix}$$

$$p(x,y) = \sum_{i=0}^{3} \sum_{j=0}^{3} a_{ij} x^i y^j$$

Physics Engine

Higher order Differential Equation Solver



Fourth-order Runge-Kutta method

$$k_{i,1} = h_i f(t_i, w_i);$$

$$k_{i,2} = h_i f(t_i + \tfrac{1}{3}h_i, w_i + \tfrac{1}{3}k_{i,1});$$

$$k_{i,3} = h_i f(t_i + \tfrac{2}{3}h_i, w_i - \tfrac{1}{3}k_{i,1} + k_{i,2});$$

$$k_{i,4} = h_i f(t_i + h_i, w_i + k_{i,1} - k_{i,2} + k_{i,3});$$

$$w_{i+1} = w_i + \tfrac{1}{8}(k_{i,1} + 3k_{i,2} + 3k_{i,3} + k_{i,4})$$

High Order Differential Equation Solver

Advantages:

- -simple and easy to implement
- -flexible step size
- -better than euler
- (-suited for bootstrapping in multistep methods)

Disadvantages:

- -single step method
    - -evaluation uses only one previous value
- -computation time

Artificial Intelligence

A* Algorithm (Artificial Intelligence)

Heuristic function: f(x) = g(x) + h(x)

- g(x) - actual distance to the current element

Euclidean distance heuristic:

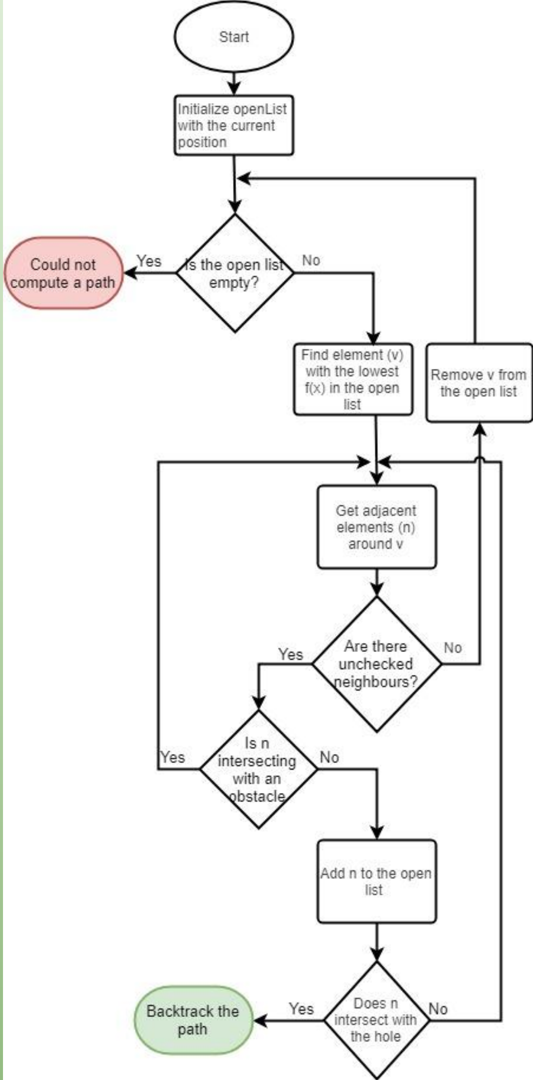
- h(x) - euclidean distance to the hole

Minimum stroke heuristic:

- h(x) - euclidean distance to the hole * #strokes

Performing Hits:

- Adjust hit strength after each move

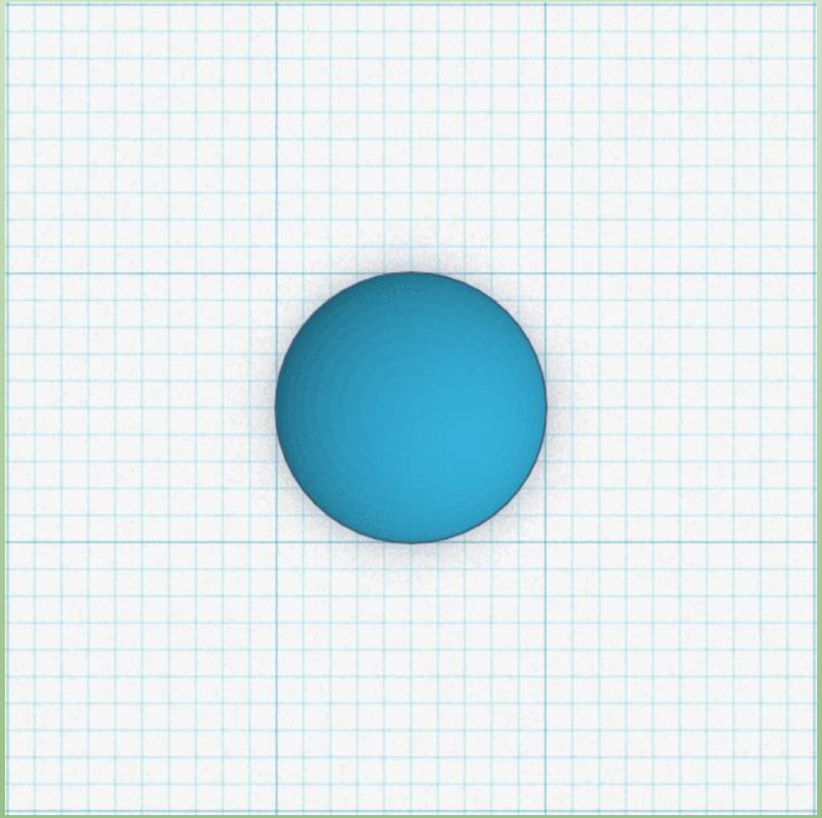

Start
Initialize openList with the current position
Is the open list empty?
Yes
Could not compute a path
No
Find element (v) with the lowest f(x) in the open list
Remove v from the open list
Get adjacent elements (n) around v
Are there unchecked neighbours?
Yes
No
Is n intersecting with an obstacle
Yes
No
Add n to the open list
Does n intersect with the hole
Yes
Backtrack the path
No

A* Algorithm - Expanding

Allows finding maze like paths

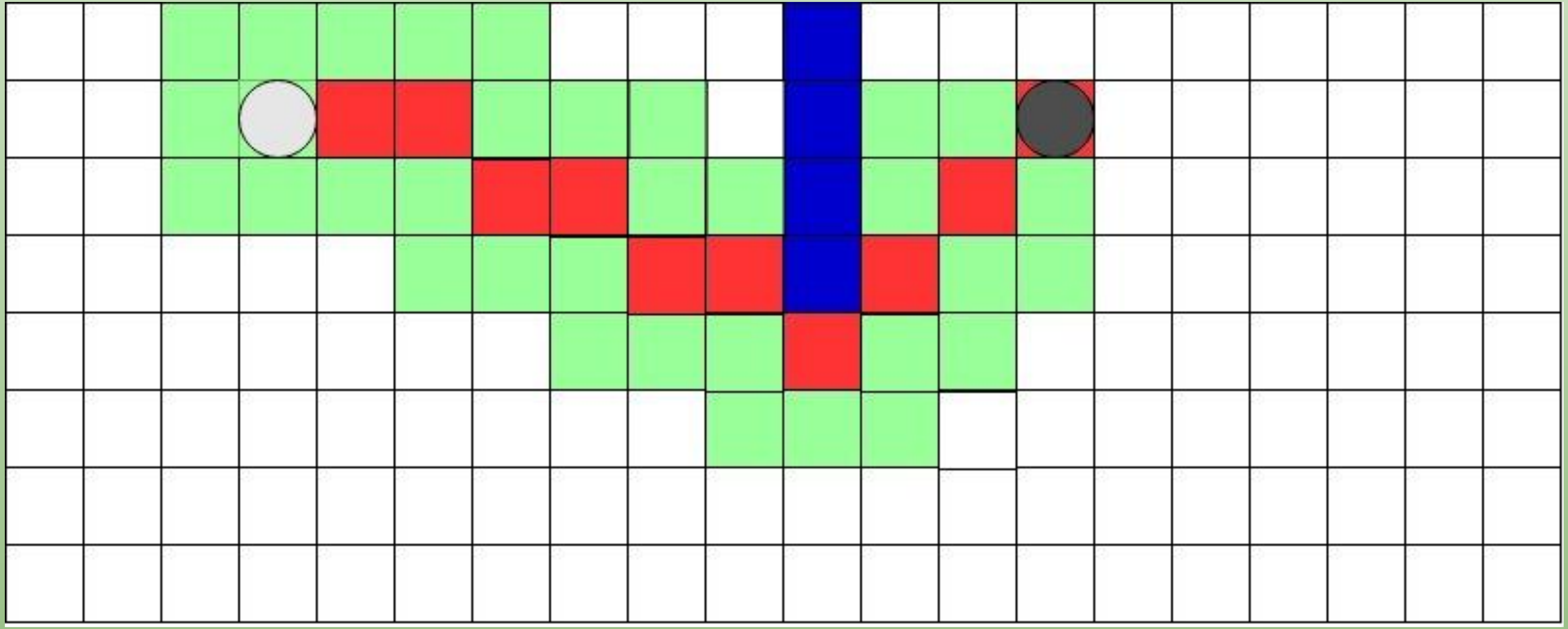
Runtime varies on the cube size

- Smaller cube size → longer runtime



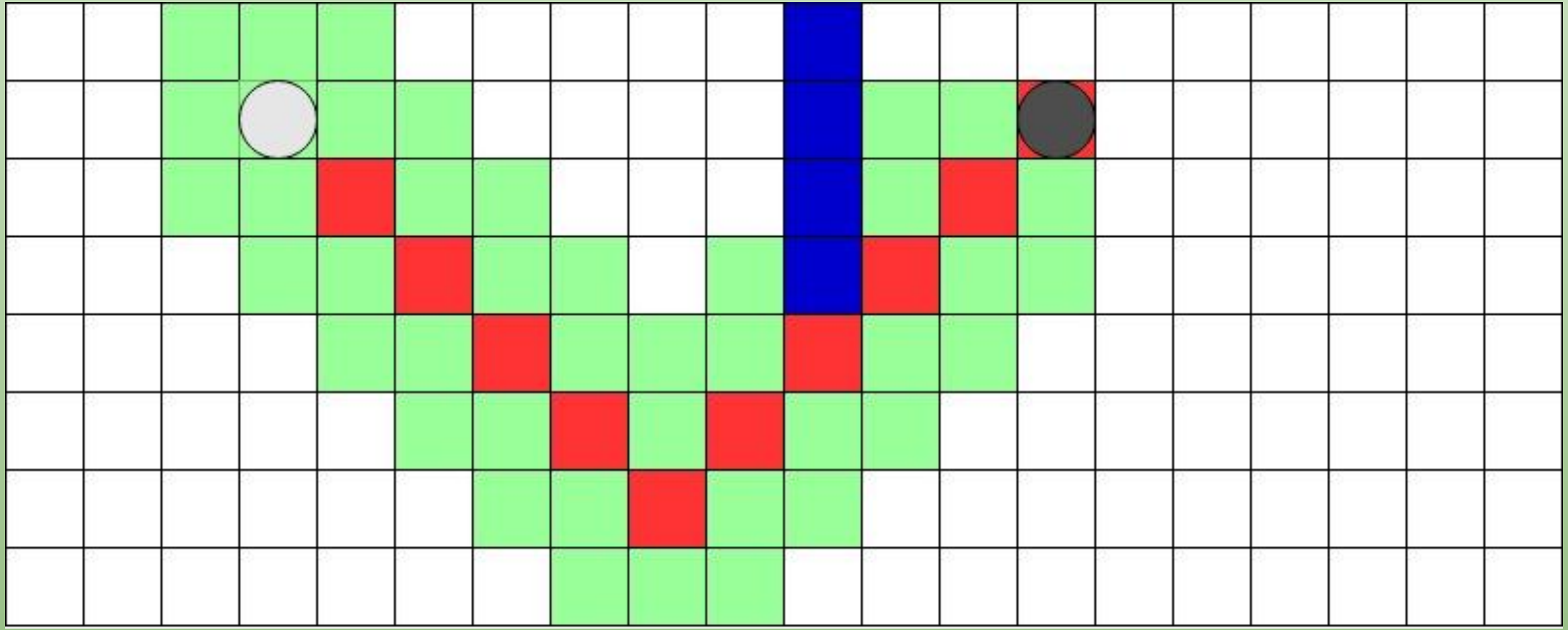
A* Algorithm - Path finding

Euclidean distance heuristic



A* Algorithm - Path finding

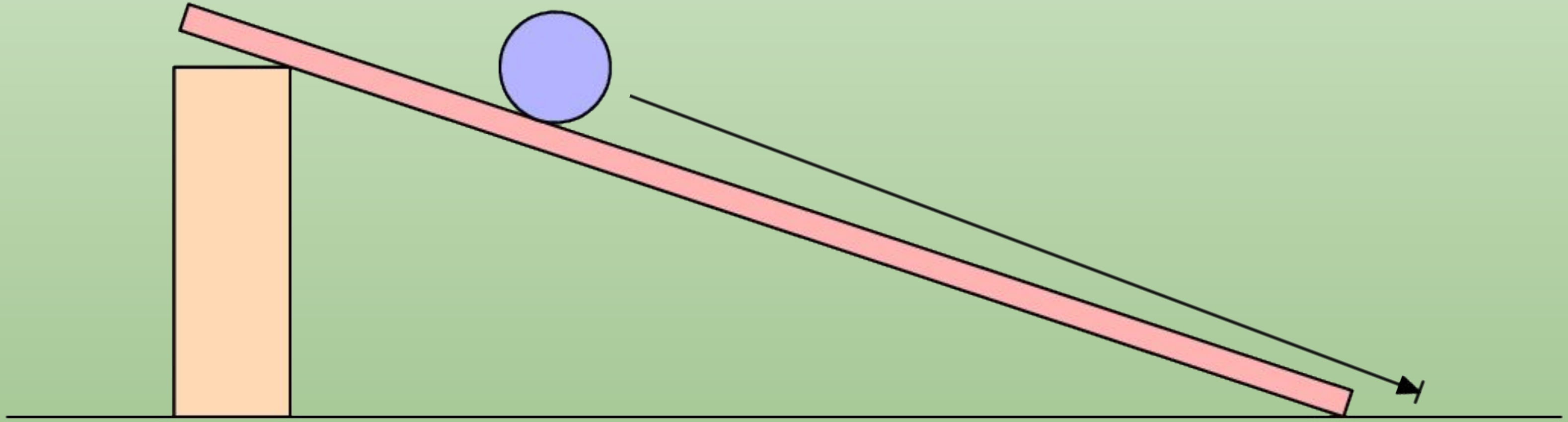
Minimum stroke heuristic



A-Star pathfinding (video)

Slopes!

Artificial Intelligence - Slopes



Cooperative Multiplayer

Multiplayer game

-allows for teams of two to play together

- each ball still has its own hole (corresponding to its colour)

-Distance constraint: balls must stay within a certain distance of each other

-positions of obstacles/ balls/ holes are chosen at random

-complete elastic collisions are enabled between balls

Entirely Elastic Collision

Other balls can be influenced and moved by the putt of another ball.

$$v_1^* = \frac{(m_1 \text{-} m_2)^* \, v_1 + 2^* m_2{}^* v_2}{m_1 + m_2}$$

$$v_2^* = \frac{(m_2 \text{-} m_1)^* \, v_2 + 2^* m_1{}^* v_1}{m_1 + m_2}$$

The one-dimensional Newtonian equation.

Multiplayer Game

Three Modes:

- -default
  - -checks distance constraint AFTER ball has naturally stopped
- -rolling
  - -checks distance constraint WHILE ball is rolling
  - -stops the ball as soon as the constraint is broken
- -pseudo elastic band
  - -checks distance constraint WHILE ball is rolling
  - -once the constraint is broken, velocities are applied to the balls

Multiplayer Mode 1 (video)

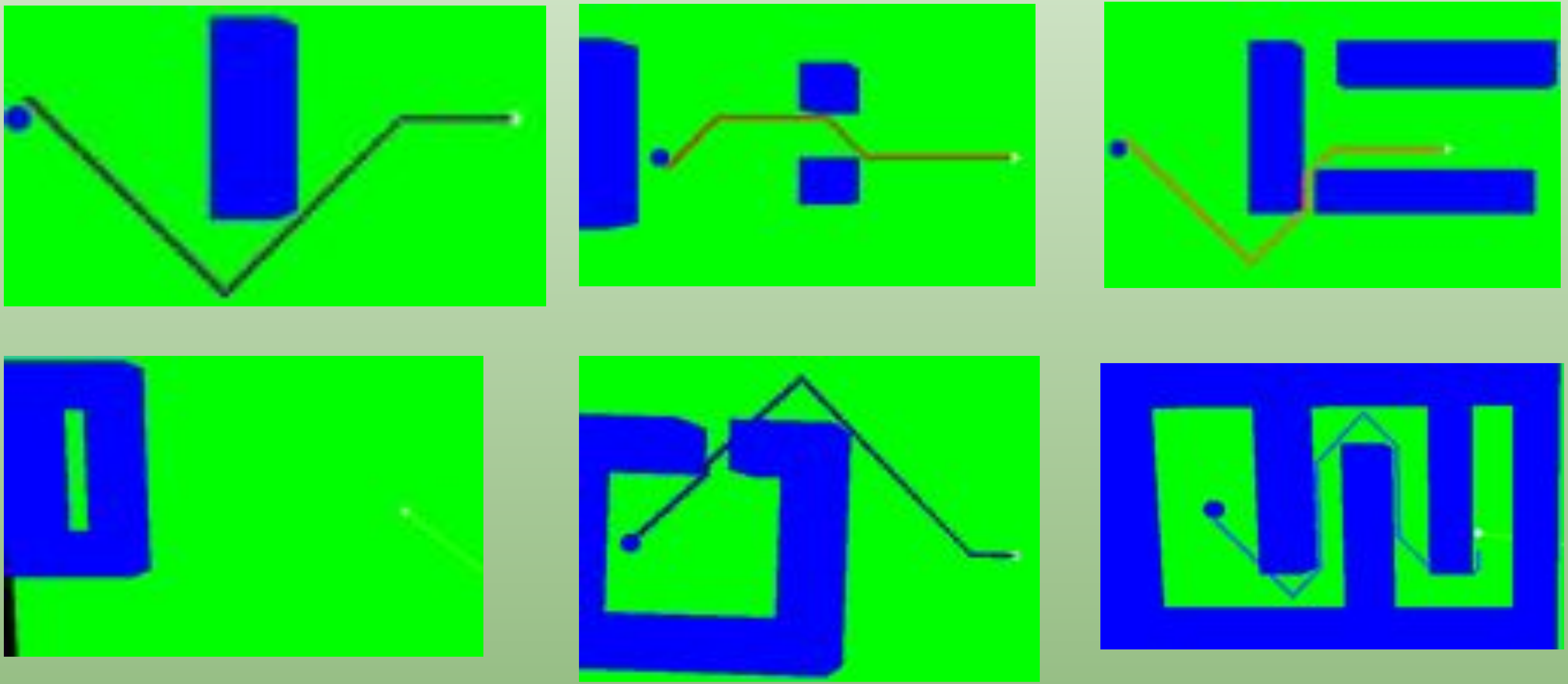
Multiplayer Mode 2 (video)

Multiplayer Mode 3 (video)

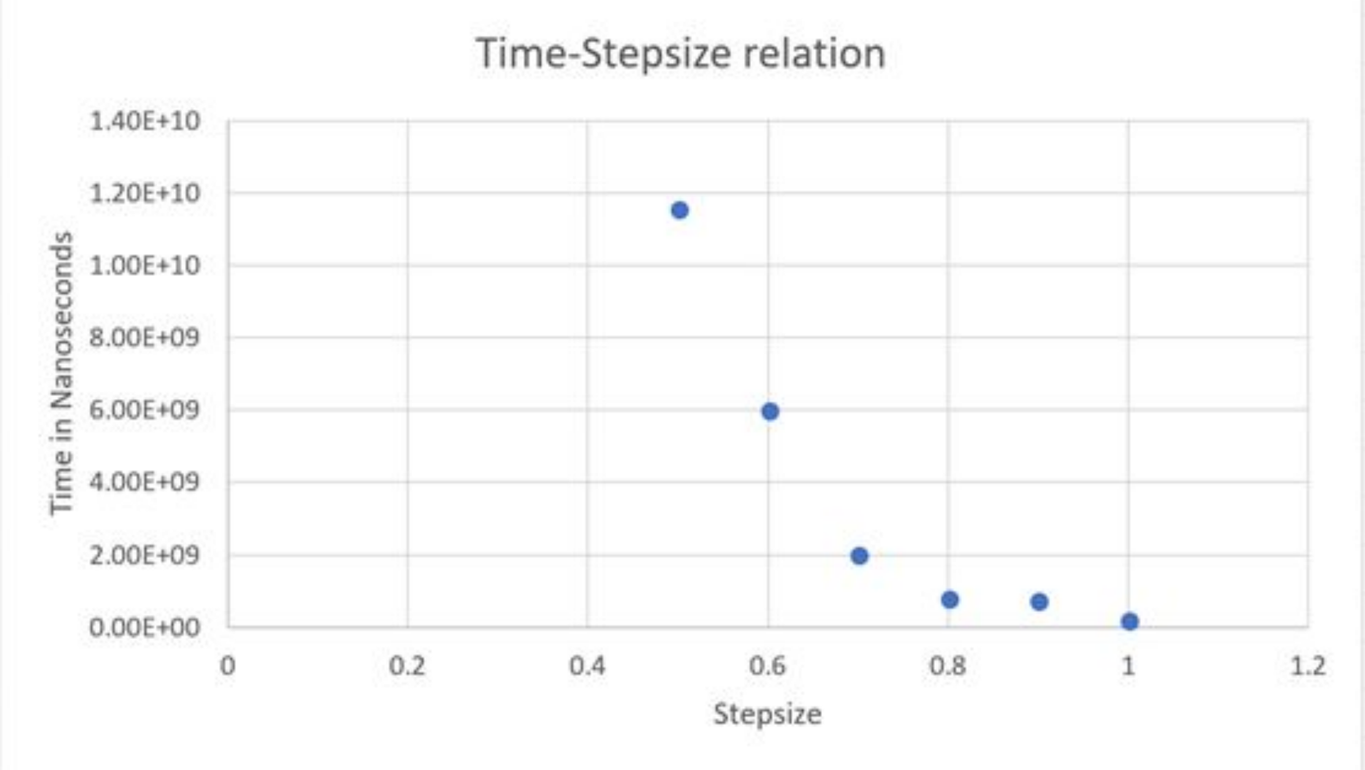
$$v_1^* = -v_1 - 0.5v_2$$
$$v_2^* = -v_2 - 0.5v_1$$

Experimentation

AI Pathfinding Test



AI step size tolerance Test



Conclusion

Thank you!