# TDT4136 Assignment 2

Fredrik Veidahl Aagaard & Karl Martin Lysø Svensby

September 2020

## Preface

### Visualizer

We implemented the visualizer component from scratch using PyGame. This allowed the A* algorithm to be performed in ”real-time” and animated. Cell state color coding is found in Table 1:

| State | Description | Color |
|---|---|---|
| STANDARD | Cell is unvisited by A* | Weighted grey[1] |
| BARRIER | Cell is not visitable by A* | Black |
| START | Cell is start cell | Red |
| GOAL | Cell is goal cell | Green |
| OPEN | Cell is in A* open set | Orange |
| CLOSED | Cell is in A* closed set | Blue |
| PATH | Cell is in A* shortest path from START to GOAL | Yellow |

Table 1: Cell states.

### Running the code

To run the code perform the following steps:

1. Set up a python3 environment
2. Unzip the provided zipfile containing the code
3. `cd A-star-Visualizer`
4. `pip install -r requirements.txt`
5. `python main.py --cell_size=n` where `n` is the desired size of each grid cell in pixels. `n` is set to 16 by default if `--cell_size` is not provided.

[1] Darker shade means higher weight on cell

## Using the application

When the pygame application has launched by running `python main.py` a blank map is initialized. Mouse/keyboard actions for the application are listed in Table 2 and 3 respecitvely. These actions can be performed on all maps. The application is fairly robust to changes during A* search. For example can maps be changed during A* search. Barriers can be drawn, but if they are drawn in already searched space, the algorithm does not recognize this. For optimal behaviour, draw/load a map and let the algorithm finish.

| `Mouse Button Press` | Response |
|---|---|
| `Left Mouse Button` | Draws standard cell |
| `Right Mouse Button` | Draws barrier cell |
| `Middle Mouse Button` | Adds/removes start/goal cell |

Table 2: Mouse button actions

| `Keyboard Press` | Response |
|---|---|
| `Space` | Starts A* algorithm |
| `1-5` | Sets map to Assignment task 1-5 |
| `0` | Sets map to blank 50x50 grid |
| `r` | Resets map to current task |
| `p` | Takes screenshot [2] |
| `Esc` | Quits application |

Table 3: Keyboard actions

## A* algorithm

The implemented A* algorithm was inspired by pseudo-code found on Wikipedia `https://en.wikipedia.org/wiki/A*_search_algorithm`.

[2] When taking a screenshot, the folder `images` is created in the current working directory, and a .png is saved there.

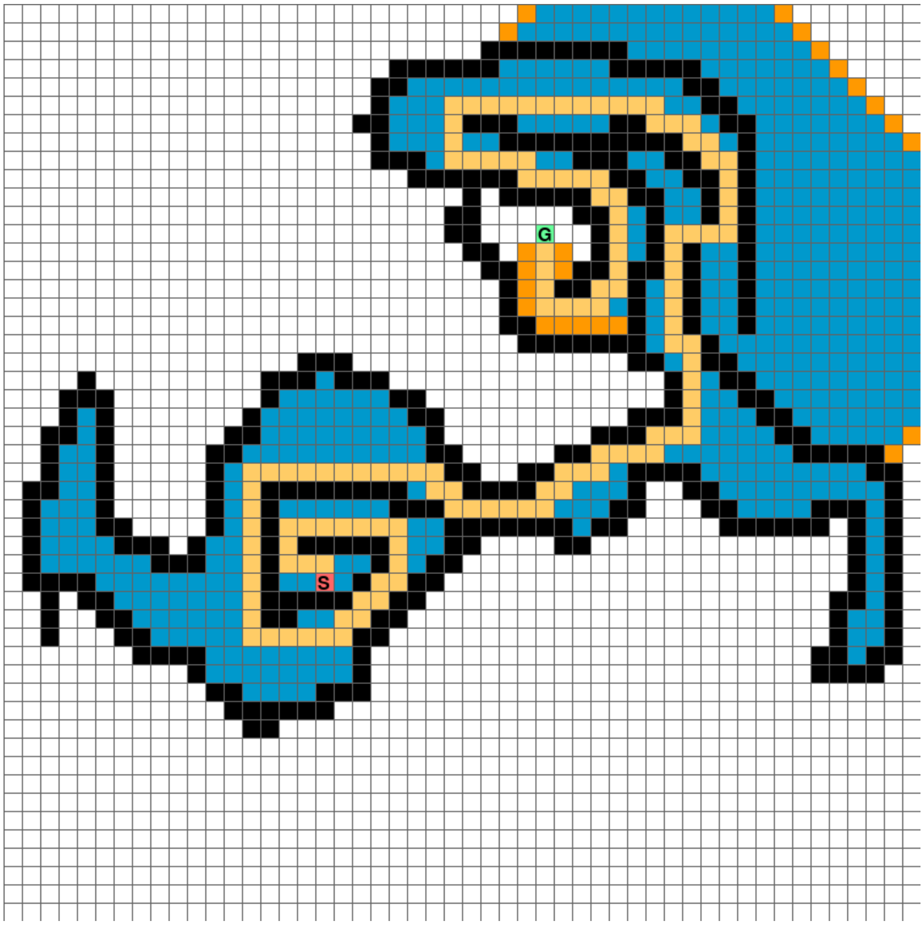

Figure 1: Example search from custom map

# 1 Part 1

## 1.1 Task 1

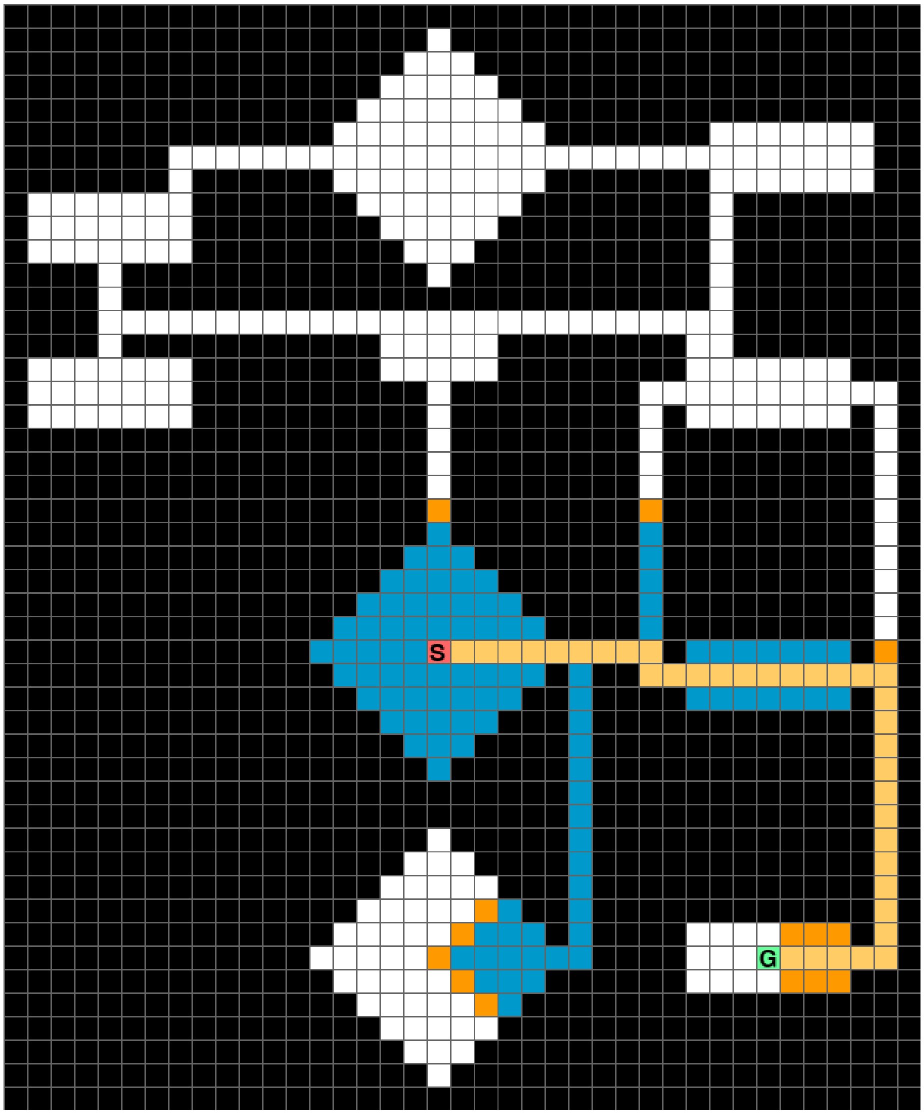

Figure 2: Task 1 visualized

### 1.2 Task 2

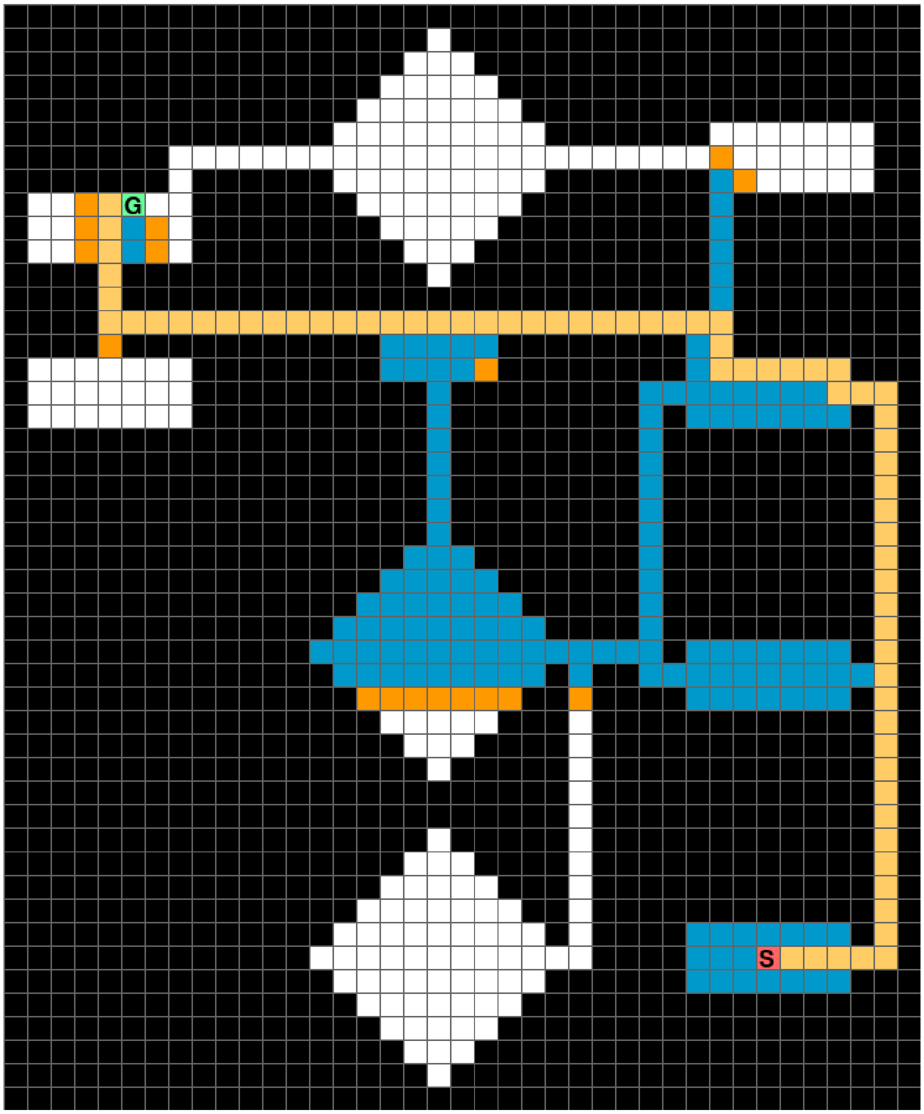

Figure 3: Task 2 visualized

# 2 Part 2

## 2.1 Task 3

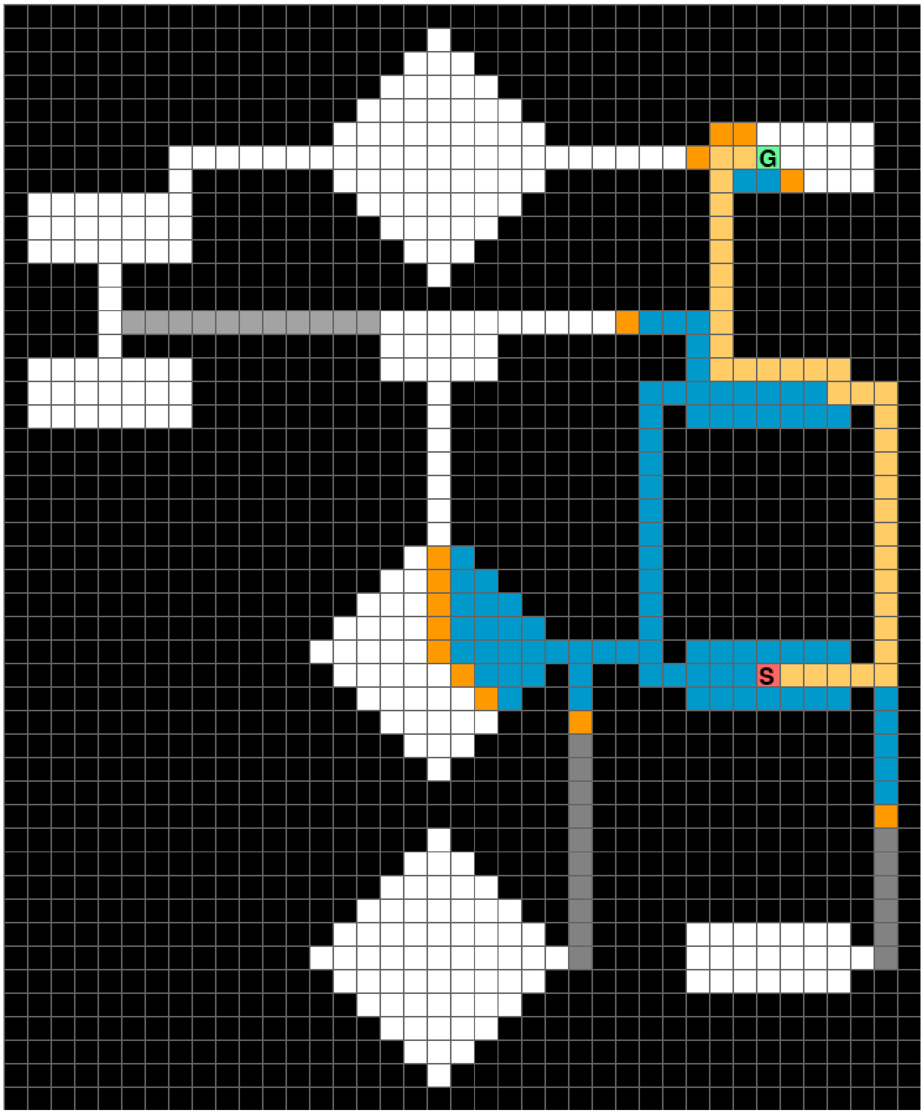

Figure 4: Task 3 visualized

## 2.2 Task 4

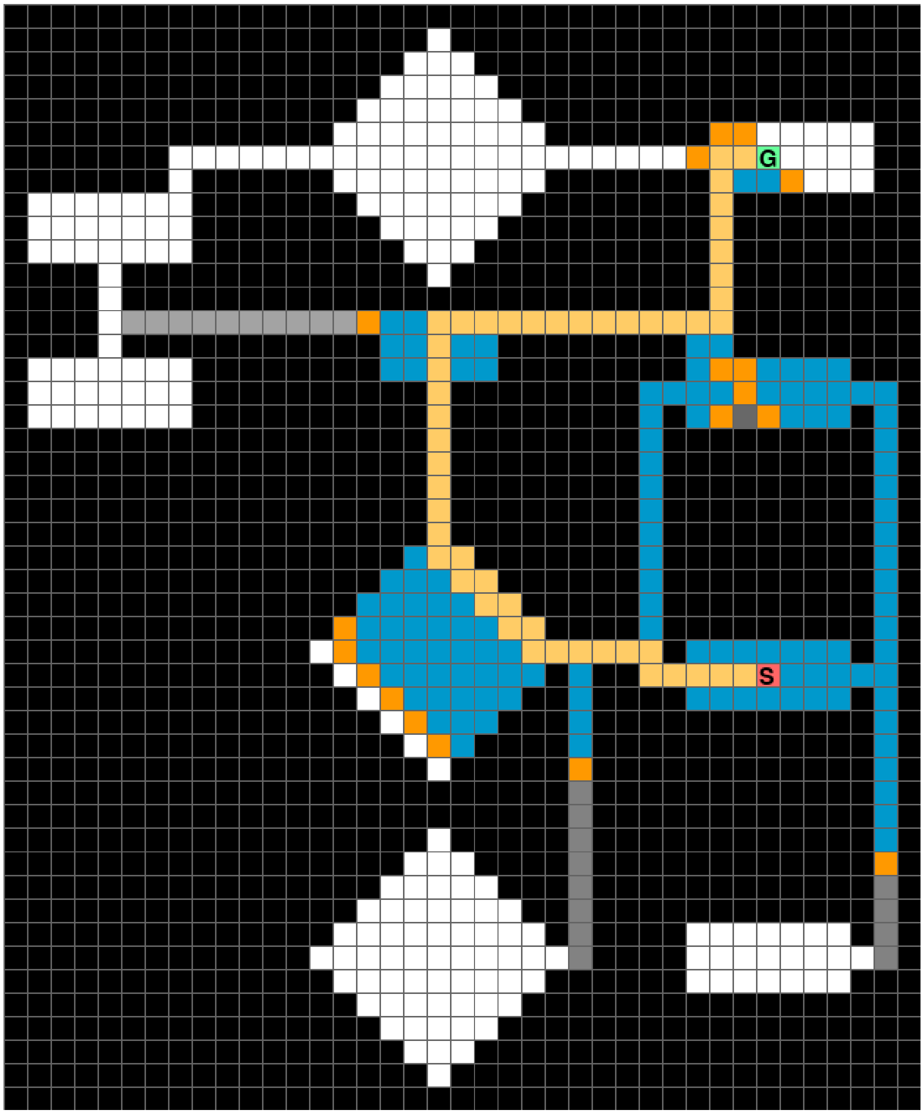

Figure 5: Task 4 visualized

# 3 Part 3

## 3.1 Task5

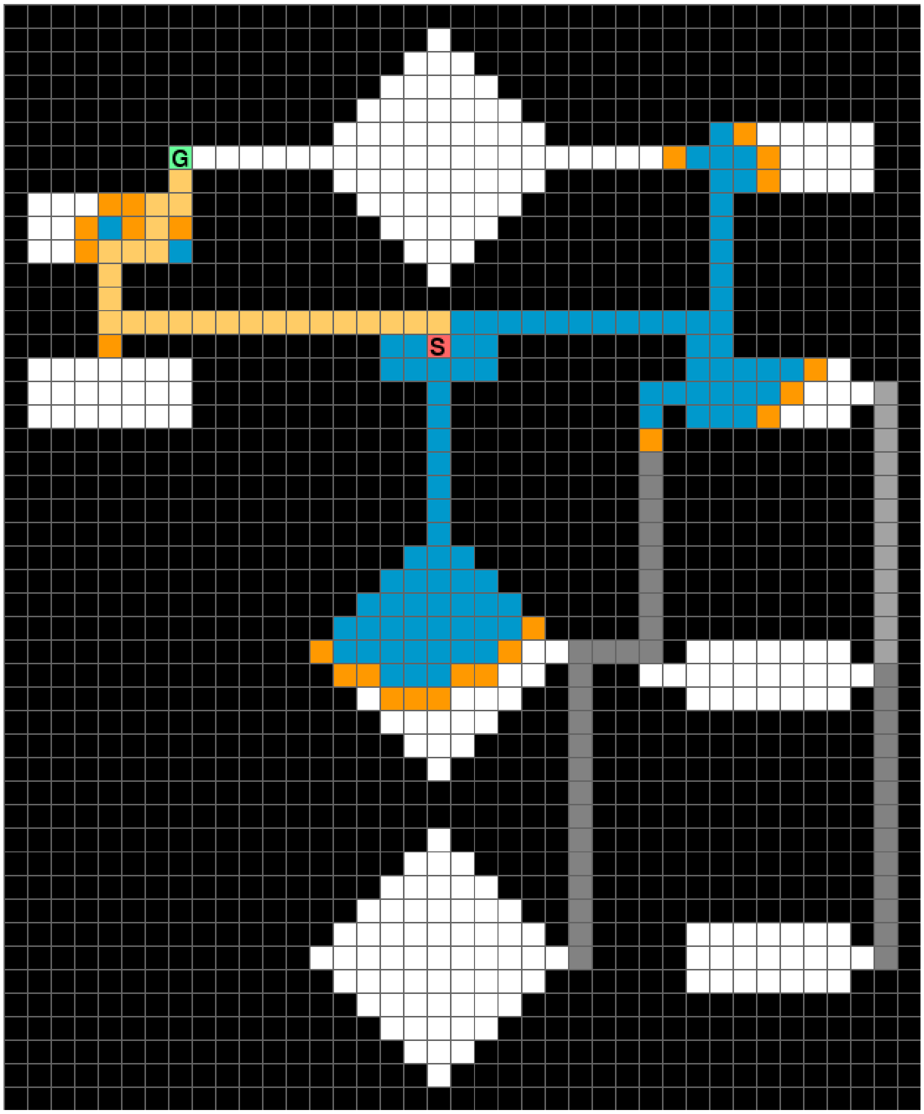

Figure 6: Task 5 visualized