

Northeastern University  
College of Engineering  
Department of Electrical and Computer Engineering  
EECE 2560: Fundamentals of Engineering Algorithms  
Fall 2024

**Project: Autonomous EV Maze Navigation**  
**Names: Vasishta Malisetty & Jani Passas**

# Introduction

## Objectives & Goals

- To allow delivery vehicle to most efficiently navigate through a maze
- To enable robot to autonomously navigate obstacles
- To develop a robust backtracking algorithm that can be applied to other fields

## Project Scope

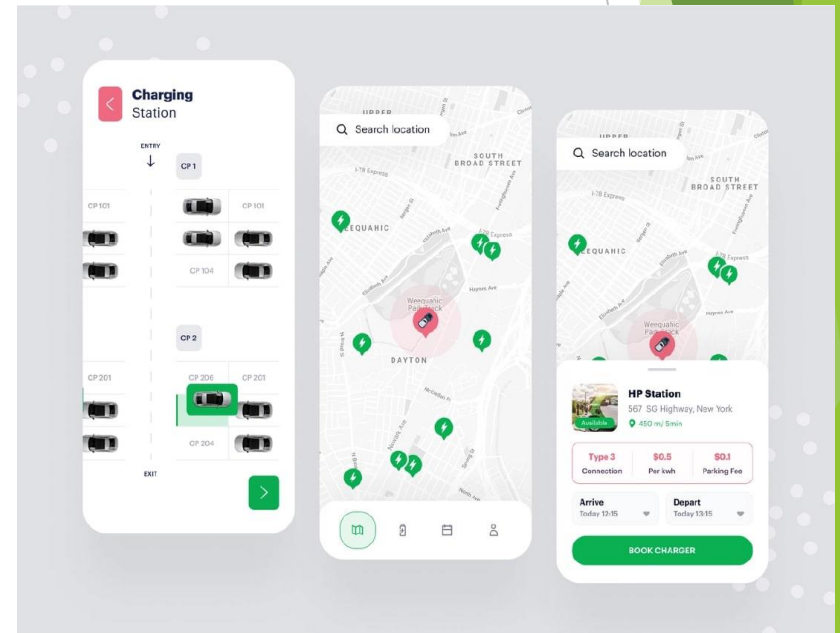
- Expand into greedy/graph algorithms by incorporating the car fueling along its path
- Incorporate the outputs onto a UI on a front-end web page

# Literature Review (Industry)



Amazon  
Delivery

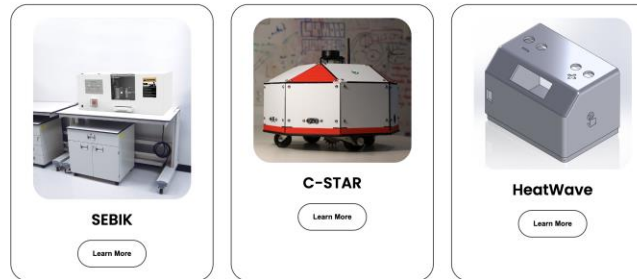
Electric  
Vehicles



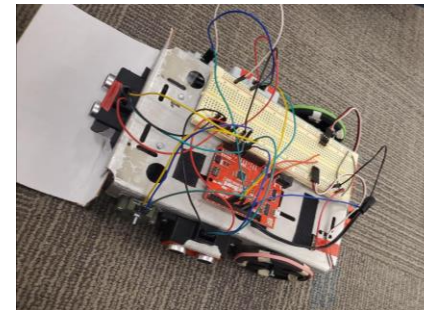
# Literature Review (Personal)

HTML &  
CSS

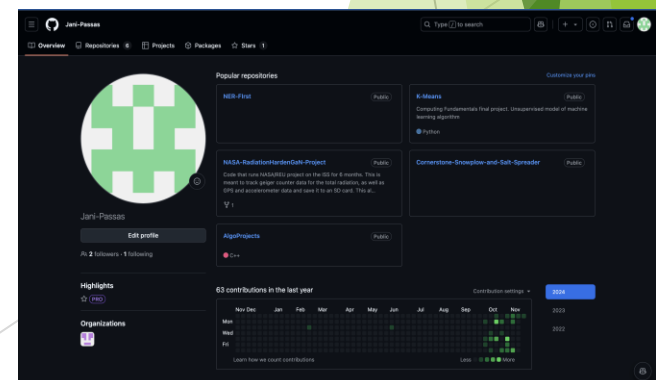
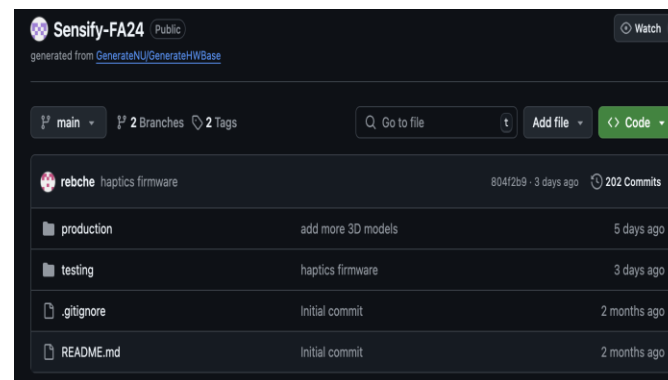
Recent  
Projects



C++



Git



# Data Structures

## Structs

- Path
  - All paths including pathTaken
- Node
  - Weight and x, y coordinates. Used for priority queue

## Vectors

- 2D matrix of maze
- Used for path tracking
- Vector pair used for directions

## Queues

- Priority queue uses min-heap and greedy to always select smallest accumulated weight

# Methodology (2 approaches)

## Dynamic Programming

Recursively finds all paths

Then picks final path with minimum total weight

## Greedy

Prim's MST and Dijkstra's inspired

- Use priority queue/min-heap
- Tree vs start and end

Selects smallest current cumulative weight

User Inputs

Maze, start/end, fuel

Dynamic or Greedy Approach

Includes charging stations and obstacles

Output

Terminal, CSV, and UI

# Back-end Overview

## IsValid(x, y, maze)

- Check boundary constraints and if cell is an obstacle
- Returns true if valid, otherwise false
- isValid:  $O(1)$

## HasAdjacentCharger(maze, x, y, rows, cols)

- Check possible moves (up, left, down, right)
- If neighbors are within bounds and is a charger, return true
- hasAdjacentCharger:  $O(1)$

## PrintPath(finalPath, maze)

- For each cell in matrix  
If it is in path, print "1"  
If it's a charger on path or neighbor, print "C"  
If it's an unused charger, print "c"  
Else, print "0"
- printPath:  $O(n*m)$

## PathToCSV(pathTaken, filename)

- Open file with filename
- Iterate through rows and write the row into the file (comma separated)
- Close the file
- PathToCSV:  $O(n*m)$

# Back-end Overview (cont)

FindAllPaths(maze, x, y, currentPath, allPaths, curWeight, endX, endY)

- If out of bounds, return
- If destination, add path to allPaths with weight
- If cell already visited, return
- For each direction  
if newCell isValid, recursively call findAllPaths to explore deeper into path until destination reached or hit dead end
- Time complexity:  $O(4^{(n*m)})$

FindMinPath(allPaths)

- Initialize minPath
- For each path in allPaths, if path weight is less than minPath, then minPath = current path
- Return minPath
- Time complexity:  $O(n)$

findGreedyPath(maze, startX, startY, endX, endY, greedyPath, fuel, maxFuel, fuelExhaustion)

- Initialize visited array, parent array, usedChargers array, and priority queue
- While priority queue not empty
  - Dequeue node with smallest weight
  - If destination, trace back to start using parent array
    - Mark path in greedyPath and return total weight
  - for each direction, move and check isValid and not visited.
    - Update weight, visit status, ad parent information.
    - Enqueue new cell into priority queue
- If destination not reached, return -1
- Time complexity:  $O((n*m) \log(n*m))$



## Analysis and Results (Backend)

# Dynamic Programming

- $O(n \cdot 4^{(n \cdot m)})$

## Greedy Algorithm

- $O((n*m) \log(n*m))$

The image shows a C++ program in a text editor and a macOS memory warning dialog. The C++ code defines a 2D vector field and a function to find the minimum value in a 2D array. The memory warning dialog indicates that the system has run out of application memory and lists the memory usage of various applications.

```

1  Homework > C++ SampleRobotNavTesting.cpp > @ main()
2  93 void pathToCSV(vector<vector<int>>& pathTaken, string filename){
3
4      103         outFile << " ";
5      104     }
6      105     }
7      106     outFile << "\\n";
8      107 }
9      108 outFile.close();
10     109 cout << "Data saved into CSV file and written to: " << filename << endl;
11     110 }
12
13 121 int main(){
14     122     //Make to traverse, 0s are obstacles
15     123     vector<vector<int>> maze = {
16         124         {1, 0, 2, 3, 1, 3, 5, 2, 2, 1},
17         125         {2, 3, 1, 0, 2, 0, 2, 5, 3, 2},
18         126         {0, 2, 6, 4, 3, 4, 3, 2, 5, 1},
19         127         {1, 7, 2, 0, 1, 6, 4, 2, 1, 3},
20         128         // {1, 0, 2, 3, 1, 0, 0, 2, 3, 2},
21         129         {2, 3, 1, 0, 2, 1, 7, 2, 6, 1},
22         130         {0, 2, 6, 4, 3, 0, 2, 2, 4, 5},
23         131         {1, 7, 2, 0, 1, 1, 5, 2, 1, 0}
24     };
25
26     132 for(size_t i=0; i=maze.size(); i++){
27         133     for(size_t j=0; j=maze[i].size(); j++){
28             134         cout << maze[i][j] << " ";
29     }
30 }
31
32 135
33 136
34 137
35 138
36 139
37 140
38 141
39 142
40 143
41 144
42 145
43 146
44 147
45 148
46 149
47 150
48 151
49 152
50 153
51 154
52 155
53 156
54 157
55 158
56 159
57 160
58 161
59 162
60 163
61 164
62 165
63 166
64 167
65 168
66 169
67 170
68 171
69 172
70 173
71 174
72 175
73 176
74 177
75 178
76 179
77 180
78 181
79 182
80 183
81 184
82 185
83 186
84 187
85 188
86 189
87 190
88 191
89 192
90 193
91 194
92 195
93 196
94 197
95 198
96 199
97 200
98 201
99 202
100 203
101 204
102 205
103 206
104 207
105 208
106 209
107 210
108 211
109 212
110 213
111 214
112 215
113 216
114 217
115 218
116 219
117 220
118 221
119 222
120 223
121 224
122 225
123 226
124 227
125 228
126 229
127 230
128 231
129 232
130 233
131 234
132 235
133 236
134 237
135 238
136 239
137 240
138 241
139 242
140 243
141 244
142 245
143 246
144 247
145 248
146 249
147 250
148 251
149 252
150 253
151 254
152 255
153 256
154 257
155 258
156 259
157 260
158 261
159 262
160 263
161 264
162 265
163 266
164 267
165 268
166 269
167 270
168 271
169 272
170 273
171 274
172 275
173 276
174 277
175 278
176 279
177 280
178 281
179 282
180 283
181 284
182 285
183 286
184 287
185 288
186 289
187 290
188 291
189 292
190 293
191 294
192 295
193 296
194 297
195 298
196 299
197 300
198 301
199 302
200 303
201 304
202 305
203 306
204 307
205 308
206 309
207 310
208 311
209 312
210 313
211 314
212 315
213 316
214 317
215 318
216 319
217 320
218 321
219 322
220 323
221 324
222 325
223 326
224 327
225 328
226 329
227 330
228 331
229 332
230 333
231 334
232 335
233 336
234 337
235 338
236 339
237 340
238 341
239 342
240 343
241 344
242 345
243 346
244 347
245 348
246 349
247 350
248 351
249 352
250 353
251 354
252 355
253 356
254 357
255 358
256 359
257 360
258 361
259 362
260 363
261 364
262 365
263 366
264 367
265 368
266 369
267 370
268 371
269 372
270 373
271 374
272 375
273 376
274 377
275 378
276 379
277 380
278 381
279 382
280 383
281 384
282 385
283 386
284 387
285 388
286 389
287 390
288 391
289 392
290 393
291 394
292 395
293 396
294 397
295 398
296 399
297 400
298 401
299 402
300 403
301 404
302 405
303 406
304 407
305 408
306 409
307 410
308 411
309 412
310 413
311 414
312 415
313 416
314 417
315 418
316 419
317 420
318 421
319 422
320 423
321 424
322 425
323 426
324 427
325 428
326 429
327 430
328 431
329 432
330 433
331 434
332 435
333 436
334 437
335 438
336 439
337 440
338 441
339 442
340 443
341 444
342 445
343 446
344 447
345 448
346 449
347 450
348 451
349 452
350 453
351 454
352 455
353 456
354 457
355 458
356 459
357 460
358 461
359 462
360 463
361 464
362 465
363 466
364 467
365 468
366 469
367 470
368 471
369 472
370 473
371 474
372 475
373 476
374 477
375 478
376 479
377 480
378 481
379 482
380 483
381 484
382 485
383 486
384 487
385 488
386 489
387 490
388 491
389 492
390 493
391 494
392 495
393 496
394 497
395 498
396 499
397 500
398 501
399 502
400 503
401 504
402 505
403 506
404 507
405 508
406 509
407 510
408 511
409 512
410 513
411 514
412 515
413 516
414 517
415 518
416 519
417 520
418 521
419 522
420 523
421 524
422 525
423 526
424 527
425 528
426 529
427 530
428 531
429 532
430 533
431 534
432 535
433 536
434 537
435 538
436 539
437 540
438 541
439 542
440 543
441 544
442 545
443 546
444 547
445 548
446 549
447 550
448 551
449 552
450 553
451 554
452 555
453 556
454 557
455 558
456 559
457 560
458 561

```

```

❏ TERMINAL
❏ janipassas@Janis-MacBook-Pro Homework % ./RobotNavCharge
1 0 2 3 1 3 2 -1 2 1 4 6 1 3
2 3 1 0 2 8 2 5 3 2 3 2 0 2
0 2 6 4 3 4 1 2 5 -1 3 2 3 0
1 7 2 0 1 6 4 2 1 3 6 2 5 3
1 0 2 3 1 0 0 2 3 2 3 1 3 5
2 3 1 0 6 -1 7 2 6 1 4 5 2 2
0 2 6 4 3 0 2 2 4 5 4 2 3 1
1 7 2 0 1 1 5 2 1 0 0 0 2 1
2 3 1 0 2 8 2 5 3 2 0 1 1 3
0 2 6 4 3 4 3 2 5 1 -1 3 4 5
1 7 2 -1 1 6 4 2 1 3 4 2 1 7
1 -1 2 0 1 0 -1 2 3 2 4 2 3 2
2 3 1 0 2 1 7 2 6 1 3 3 2 1
0 2 6 4 3 0 2 2 4 5 0 0 2 1

Boundaries for the input are 13 and 13
Please enter start coordinates (x y): 1 1
Please enter end coordinates (x y): 12 12
Please enter starting fuel: 15
Please enter max fuel capacity: 30
Please enter 1 for dynamic approach and 2 for greedy approach: 2
Minimum path weight: 49
Path traveled was:
0 0 1 1 1 1 1 C 0 0 0 0 0 0
0 1 1 0 0 0 1 0 0 0 0 0 0 0
0 0 0 0 0 0 0 1 1 0 C 0 0 0 0
0 0 0 0 0 0 0 0 1 0 0 0 0 0
0 0 0 0 0 0 0 0 1 0 0 0 0 0
0 0 0 0 0 0 C 0 1 0 0 0 0 0 0
0 0 0 0 0 0 0 0 1 0 0 0 0 0
0 0 0 0 0 0 0 0 1 1 0 0 0 0
0 0 0 0 0 0 0 0 0 1 C 1 0 0
0 0 0 C 0 0 0 0 0 0 0 1 1 0
0 C 0 0 0 0 C 0 0 0 0 0 1 0
0 0 0 0 0 0 0 0 0 0 0 0 1 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0

Remaining fuel: 19
Data saved into CSV file and written to: output.csv
❏ janipassas@Janis-MacBook-Pro Homework %

```

[illegible]

# CSV

# Frontend Overview

## LoadCSVFile()

- Fetch CSV file
- Parse the CSV files to get rows for the UI grid
- Time Complexity:  $O(n)$

## RenderGrid()

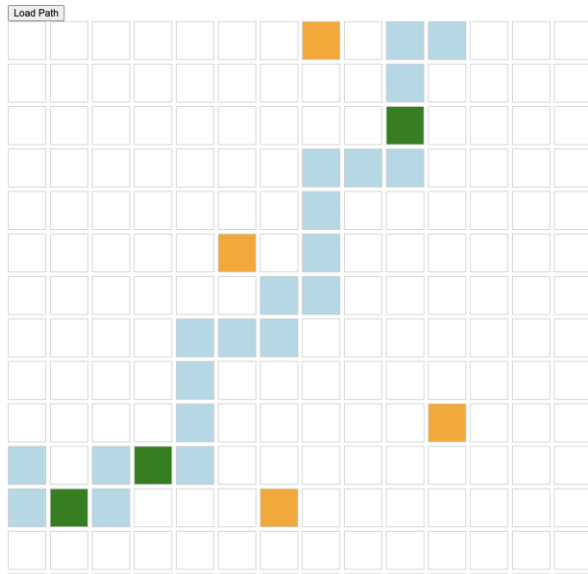
- If `grid == 1`, add block to path
- Else, do not add block to the path
- Time Complexity:  $O(n^2)$

# Analysis and Results (Frontend)

UI

HTML

## Backtracking UI



```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta http-equiv="X-UA-Compatible" content="IE=edge">
6   <meta name="viewport" content="width=device-width, initial-scale=1.0">
7   <title>Pathfinding Maze Solver</title>
8   <style>
9     #gridContainer {
10       display: grid;
11       grid-template-columns: repeat(5, 40px);
12       gap: 2px;
13     }
14     .grid-cell {
15       width: 50px;
16       height: 50px;
17       border: 1px solid black;
18       display: flex;
19       justify-content: center;
20       align-items: center;
21       cursor: pointer;
22     }
23     .path-cell {
24       background-color: yellow;
25     }
26     .obstacle-cell {
27       background-color: grey;
28     }
29   </style>
30 </head>
31 <body>
32   <h1>Pathfinding Maze Solver</h1>
33   <div id="grid" class="grid"></div>
34   <button id="solveBtn">Find Shortest Path</button>
35 </body>
```

Server

```
vmalisetty_23@Vasishtas-MacBook-Pro AlgoProjects % python3 -m http.server
Serving HTTP on :: port 8000 (http://[::]:8000/) ...
```

# Discussion

## Limitations

Based on an Ideal  
Environment

Memory space

## Applications

Energy Efficient Travel

Roomba/warehouse/delivery  
robotics

# Conclusion

## Conclusions



Greedy > Dynamic  
Programming



Successfully connected  
Frontend & Backend

## Future Steps



Develop fully  
responsive UI

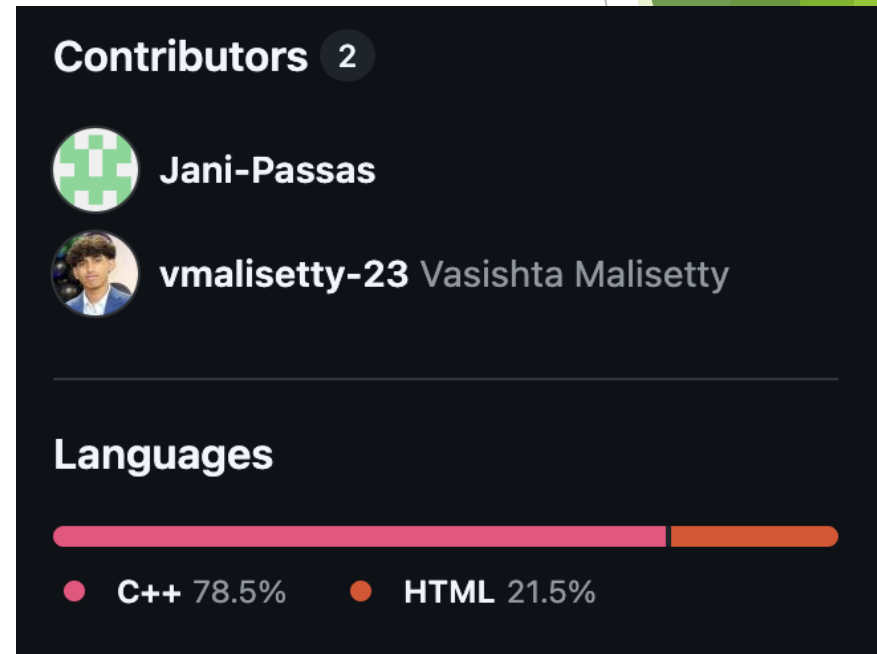


Add charging stations  
for dynamic

# Analysis and Results (Git)

Commits

Codebase



# References

- ▶ [https://www.amazon.science/latest-news/the-science-behind-grouping-amazon-package-deliveries#:~:text=delivered%20on%20time.-,Customer%20Order%20and%20Network%20Density%20OptimizeR%20\(CONDOR\)%20has%20led%20to,to%20identify%20effective%20delivery%20options.](https://www.amazon.science/latest-news/the-science-behind-grouping-amazon-package-deliveries#:~:text=delivered%20on%20time.-,Customer%20Order%20and%20Network%20Density%20OptimizeR%20(CONDOR)%20has%20led%20to,to%20identify%20effective%20delivery%20options.)
- ▶ <https://www.sciencedirect.com/science/article/pii/S0142061524001522#:~:text=This%20method%20refers%20to%20the,current%20research%20presented%20in%20Ref>
- ▶ <https://www.geeksforgeeks.org/prims-minimum-spanning-tree-mst-greedy-algo-5/>
- ▶ <https://www.geeksforgeeks.org/rat-in-a-maze/>
- ▶ <https://www.geeksforgeeks.org/introduction-to-dijkstras-shortest-path-algorithm/>

# Thank You for Listening!

## Names

- Vasishta Malisetty
- Jani Passas

## Project Name

- Autonomous EV Maze Navigation