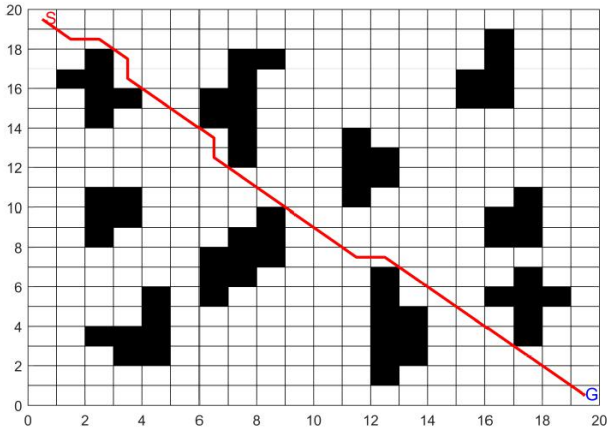
BAB II

LANDASAN TEORI

2.1 Path Planning

Path planning atau perencanaan jalur merupakan suatu metode dalam melakukan rancangan jalur optimal dari suatu titik ke titik lainnya. Pada *path planning* akan mengoptimalkan jarak jalur dari sumber dan tujuan sehingga mendapatkan jalur yang optimal[2], [13]. Sehingga dengan hasil jalur yang optimal serta waktu pencarian yang cepat akan membuat program menjadi efektif.

Pada perencanaan jalur terdapat titik *start* dan titik *goal*. Hasil dari pencarian jalur akan mendapatkan jalur dari titik *start* sampai titik *goal*. Pada lingkungan yang akan dilakukan proses pencarian jalur biasanya terdapat beberapa *obstacle*. Proses pencarian jalur akan menghindari setiap *obstacle* yang ada[13], [14]. Menghindari *obstacle* akan membuat hasil pembuatan jalur tidak menabrak dengan *obstacle*. Berikut contoh *path planning* dapat dilihat pada **Gambar 2.1**.



Gambar 2.1 *Contoh Path Planning[15]*

2.2 Algoritma Breadth First Search (BFS)

Algoritma BFS merupakan suatu algoritma yang digunakan untuk melakukan proses *path planning*. Algoritma ini ditemukan oleh Konrad Zuse pada tahun 1945.

Algoritma BFS dikenal sebagai pencarian dengan mengunjungi suatu *node* kemudian mengunjungi semua *node* yang bertetangga dengan *node* tersebut[16]. Lingkungan pada proses *path planning* algoritma BFS menggunakan *grid map*.

Algoritma ini merupakan algoritma sederhana yang dimana dimulai dari titik awal hingga titik akhir. Proses pencarian titik akhir pada algoritma ini dilakukan secara bertahap sesuai dengan panjang langkah. Algoritma dimulai pada *node* awal dan memeriksa semua *node* tetangga. Algoritma memeriksa *node* tetangga apakah ada yang belum dikunjungi dan seterusnya[11].Proses penambahan *node* dilanjutkan terus menerus hingga mencapai titik akhir. Sebuah node baru bergerak ke semua arah yang diizinkan yang dapat dilalui robot, terkecuali pada arah yang terdapat *obstacle*. Berikut *pseudo-code* algoritma BFS dapat dilihat pada **Gambar 2.2**.

Algorithm 1 Breadth First Search

```
1:  function BFS(adj, s)
2:      (d, π) ← ({ s : 0 }, { s : null })
3:      frontier ← [s]
4:      while frontier ≠ [] do
5:          next ← []
6:          for u ∈ frontier do
7:              for v ∈ adj[u] do
8:                  if v ∉ d then
9:                      (d[v], π[v]) ← (d[u] + 1, u)
10:                     APPEND(next, v)
11:                 end if
12:             end for
13:         end for
14:         frontier ← next
15:     end while
16:     return (d,π)
17: end function
```

Gambar 2.2 *Pseudo-code BFS Algorithm[17]*

2.3 Algoritma A*

Algoritma A* merupakan salah satu algoritma dalam melakukan proses *path planning*. Algoritma ini diperkenalkan oleh Peter Hart, Nils Nilsson dan Bertram Raphael pada tahun 1968[18]. Algoritma ini menggunakan kombinasi pencarian

heuristic serta pencarian berdasarkan yang terpendek[10]. Berikut *pseudo-code* algoritma A* dapat dilihat pada **Gambar 2.3**.

Pseudo-code A* Algorithm

```
function  A_Star (start, goal)
openset[grid(start)]=start
while openset is not empty
      c ← min((openset.cost)+heuristic(goal,openset))
      current ← openset[c]
      if current=goal // end
      remove current from openset
      closedset ← current
      for i to 8
      node=Node(current.x + movement[i][0],
                current.y + movement[i][1],
                current.cost + movement[i][2],c)
      n=grid(node)
      if node not safe, do nothing
      if n in closed, do nothing
      if n not in openset:
         openset[n]=node
      else:
         if openset[n].cost > node.cost
         openset[n]=node
final path
end function
```

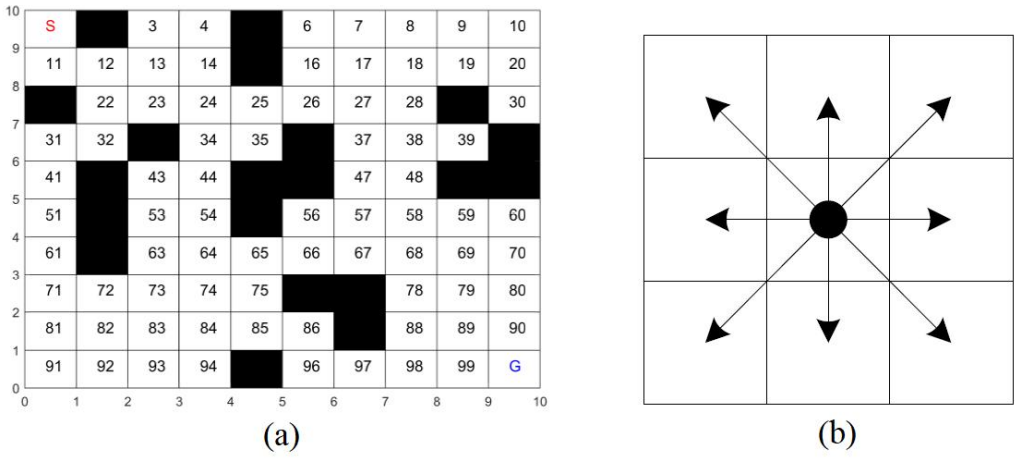
Gambar 2.3 *Pseudo-code A* Algorithm*

Algoritma A* yang menggunakan fungsi heuristik terdapat beberapa elemen kunci yaitu *open list* dan *closed list*. *Open list* pada A* dibuat masing – masing untuk menyimpan calon *node* yang akan dideteksi, dan *closed list* dibuat untuk menyimpan node yang dipilih dari jalur akhir. Sesuai dengan ukuran dari nilai tersebut, node -node di lingkungan tersebut disusun dari yang terendah ke tertinggi ke dalam *open list*[6]. Algoritma A* menggunakan notasi matematika yang dituliskan sebagai:

$$f(n) = g(n) + h(n) \quad (1)$$

Dimana *f(n)* adalah fungsi evaluasi dari simpul n saat ini, *g(n)* adalah biaya sebenarnya dari jalur dari simpul awal ke simpul saat ini n, *h(n)* adalah biaya evaluasi jalur optimal dari simpul saat ini n ke simpul tujuan g[6].

Pada masalah perencanaan jalur harus menghasilkan perencanaan yang terpendek. Pada hasil perencanaan yang terpendek akan menghasilkan rute yang optimal. Arah gerak dari A* dapat bergerak ke delapan arah. Pada A* menggunakan peta jenis yang sama dengan BFS yaitu dalam bentuk *grid map*. Contoh model lingkungan dapat dilihat pada **Gambar 2.4**.



Gambar 2.4 *Model lingkungan:(a) Contoh Grid Map (b) Kemungkinan arah gerakan[15]*