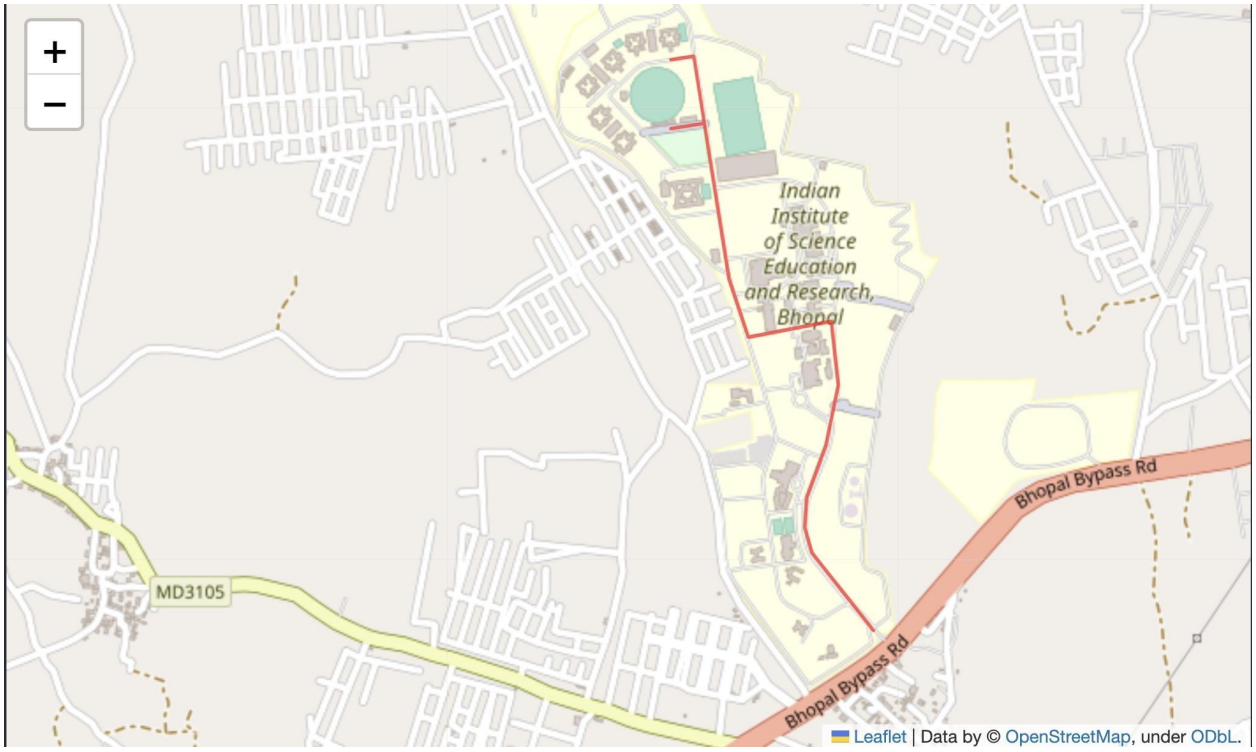
Noise-Driven Route Planning Using Genetic and A* Algorithms



Path from **Hostel - 7** to **IISERB Gate** by implementing **A* algorithm**

A* search Algorithm

A* search algorithm evaluates nodes by combining g(n), the cost to reach the node, and h(n), the cost to get from the node to the goal:

f(n) = g(n) + h(n)

Since g(n) gives the path cost from the start node to node n, and h(n) is the estimated cost of the cheapest path from n to the goal, we have -

f (n) = estimated cost of the cheapest solution through n .

Problems and challenges faced in implementing A* Search

- **While implementing , the pointer used to oscillate between nodes -**

  While calculating the g-heuristic, the actual cost in going from neighburing nodes is done by calculating distance between various geographical coordinates between the nodes. This sometimes lead to pointer oscillating between points and never reaching the destination. In order to overcome this problem, the g-heuristic is calculated by taking the straight line distace between the on-road coordinates of neighbouring nodes.

- **Incorporating noise in A* Search algorithm -**

  While calculating the **g(n)** for node **n** we have also added a noise component to it. The **noise component** is the mean of the noise amplitudes coming from the road that connects the two nodes multiplied by **100.** This is like adding another **20 metres to the distance to account for the traffic on roads**.

- **Calculating h(n) -**

  To ensure the admissibility of the A* Search algorithm, h(n) is calculated by the straight line distances between the node **n** and the destination.

Genetic Algorithm

Noise-driven route planning using genetic algorithms is a computational approach to finding the best route between two or more locations. It involves using a combination of genetic algorithms and noise to explore possible routes and optimize the solution. Genetic algorithms are a type of heuristic optimization technique inspired by natural selection and genetics. They work by creating a population of potential solutions, and then evolving these solutions over time through a process of selection, crossover, and mutation. This process mimics the natural selection process in which the fittest individuals are selected for reproduction, leading to the evolution of species over time.
In noise-driven route planning, the genetic algorithm is augmented with noise to help explore a wider range of potential routes. The noise can be in the form of random perturbations to the routes or the fitness function, which can help avoid getting stuck in local optima and lead to a more diverse and optimal set of solutions.

Over here the noise has been used in calculating the fitness function.

```
function GENETIC-ALGORITHM(population, FITNESS-FN) returns an individual
  inputs: population, a set of individuals
          FITNESS-FN, a function that measures the fitness of an individual

  repeat
    new_population ← empty set
    for i = 1 to SIZE(population) do
      x ← RANDOM-SELECTION(population, FITNESS-FN)
      y ← RANDOM-SELECTION(population, FITNESS-FN)
      child ← REPRODUCE(x, y)
      if (small random probability) then child ← MUTATE(child)
      add child to new_population
    population ← new_population
  until some individual is fit enough, or enough time has elapsed
  return the best individual in population, according to FITNESS-FN

function REPRODUCE(x, y) returns an individual
  inputs: x, y, parent individuals

  n ← LENGTH(x); c ← random number from 1 to n
  return APPEND(SUBSTRING(x, 1, c), SUBSTRING(y, c + 1, n))
```

The Pseudo Code for implementing the Genetic algorithm

The Algorithm has been implemented exactly as given in the Psuedo Code .

Incorporating Noise in the genetic algorithm

Noise is used in evaluating the cost which is the **inverse** of the fitness function. The two parameters used in calculating cost is **distance and noise.** One could assume **fitness function as the inverse of the noise.**

Stopping Criteria

The stopping criteria for the algorithm is when only **one child** is left.