

- ট্রাফিক লাইট কন্ট্রোল কোড ব্যাখ্যা (Explanation in Bengali)
  - ১. SUMO Setup (পরিবেশ প্রস্তুতি)
  - ২. ImprovedSUMOEnv ক্লাস (এনভায়রনমেন্ট)
  - ৩. ImprovedDQNAgent ক্লাস (এজেন্ট বা মস্তিষ্ক)
  - ৪. Training Loop (ট্রেনিং প্রক্রিয়া)
  - ৫. Reward (পুরস্কার) কীভাবে ক্যালকুলেট করা হয়?
  - ৬. নিউরাল নেটওয়ার্ককে কী কী ডেটা (State) দেয়া হয়?
  - ৭. ইয়েলো ও গ্রিন টাইমের মান কেন সেট করা হয়?

# ট্রাফিক লাইট কন্ট্রোল কোড ব্যাখ্যা (Explanation in Bengali)

এই কোডটি Deep Q-Learning (DQN) ব্যবহার করে একটি ট্রাফিক সিগন্যালকে বুদ্ধিমানভাবে নিয়ন্ত্রণ করার জন্য তৈরি করা হয়েছে। নিচে এর প্রধান অংশগুলো সহজভাবে বাংলায় ব্যাখ্যা করা হলো:

## ১. SUMO Setup (পরিবেশ প্রস্তুতি)

কোডের শুরুতে **SUMO\_HOME** পথ সেট করা হয়েছে এবং **traci** লাইব্রেরি ইম্পোর্ট করা হয়েছে। **traci** (Traffic Control Interface) হলো একটি টুল যা দিয়ে পাইথন কোড ব্যবহার করে SUMO সিমুলেশন নিয়ন্ত্রণ করা যায়।

## ২. ImprovedSUMOEnv ক্লাস (এনভায়রনমেন্ট)

এটি ট্রাফিক সিমুলেশনের পরিবেশ তৈরি করে। এই ক্লাসের প্রধান ফাংশনগুলো হলো:

- \_\_init\_\_(self, config\_file, use\_gui)**: এটি ক্লাসের প্রাথমিক সেটআপ করে। এখানে সিমুলেশন কমান্ড (SUMO GUI বা CLI), স্টেট সাইজ (২০টি তথ্য), অ্যাকশন সাইজ (৪টি অপশন) এবং ইয়েলো ও গ্রিন টাইমের মান সেট করা হয়।
- reset(self)**: প্রতিটি এপিসোড শুরু করার সময় এই ফাংশনটি কল করা হয়। এটি পুরনো সিমুলেশন বন্ধ করে নতুন করে ট্রাফিক সিমুলেশন শুরু করে এবং ট্রাফিক লাইটের আইডি (ID) সংগ্রহ করে। সব মিটার (যেমন: জ্যামের সময়, গাড়ির সংখ্যা) শূন্য থেকে শুরু হয়।

- **\_get\_state(self)**: এটি নিউরাল নেটওয়ার্কের ইনপুট তৈরি করে। প্রতিটি লেন থেকে গাড়ির সংখ্যা, অপেক্ষার সময় এবং গড় গতিবেগের ডেটা নিয়ে একটি ভেক্টর (Array) তৈরি করে। যদি ২০টির কম তথ্য থাকে, তবে বাকিগুলো শূন্য (Zero Padding) দিয়ে পূর্ণ করা হয়।
- **step(self, action)**: এটি সবচেয়ে গুরুত্বপূর্ণ ফাংশন। এটি AI-এর নেয়া সিদ্ধান্ত (Action) কার্যকর করে। যদি সিগন্যাল বদলাতে হয়, তবে প্রথমে ইয়েলো লাইট জ্বালায় এবং পরে গ্রিন লাইট সেট করে। এরপর সিমুলেশন কিছু সময় চালায় এবং নতুন অবস্থা (State), রিওয়ার্ড (Reward) ও সিমুলেশন শেষ হয়েছে কিনা (Done) তা রিটার্ন করে।
- **\_get\_metrics(self)**: এটি রিওয়ার্ড হিসেব করার জন্য সাহায্যকারী ফাংশন। এটি ট্রাফিক লাইটের আওতাধীন সব লেন থেকে মোট অপেক্ষার সময় এবং জ্যামে দাঁড়িয়ে থাকা গাড়ির সংখ্যা খুঁজে বের করে।
- **\_get\_veh\_count(self)**: এটি বর্তমানে রাস্তায় মোট ঠিক কতগুলো গাড়ি চলাচল করছে তা গুনে বের করে। এটি সিমুলেশনের পারফরম্যান্স ট্র্যাক করতে ব্যবহৃত হয়।

## ৩. ImprovedDQNAgent ক্লাস (এজেন্ট বা মস্তিষ্ক)

এটি হলো ট্রাফিক সিগন্যালের 'মস্তিষ্ক' যা অভিজ্ঞতা থেকে শেখে।

- **\_build\_model()**: এটি একটি নিউরাল নেটওয়ার্ক তৈরি করে যার ইনপুট হলো রাস্তার অবস্থা এবং আউটপুট হলো ৪টি সম্ভাব্য অ্যাকশনের ভ্যালু (Q-values)।
- **act()**: এটি রাস্তার অবস্থা দেখে সিদ্ধান্ত নেয়। শুরুতে এটি এলোমেলোভাবে (Random) সিগন্যাল বদলায় (Exploration), কিন্তু ধীরে ধীরে এটি অভিজ্ঞতা থেকে সেরা সিদ্ধান্ত নেয়া শুরু করে।
- **replay()**: এটি 'Experience Replay' মেমোরি থেকে পুরনো অভিজ্ঞতা নিয়ে নিজেকে আপডেট করে (Backpropagation)।
- **Target Network**: শেখার প্রক্রিয়াকে আরও স্থিতিশীল করতে এখানে দুটি নেটওয়ার্ক ব্যবহার করা হয়েছে। একটি নেটওয়ার্ক সবসময় শেখে, আর মাঝে মাঝে অন্য নেটওয়ার্কটি (Target) আপডেট করা হয়।

## ৪. Training Loop (ট্রেনিং প্রক্রিয়া)

**train()** ফাংশনটি নির্দিষ্ট সংখ্যক এপিসোড (যেমন: ৩০ বার) সিমুলেশন রান করে। প্রতিবার AI সিগন্যাল নিয়ন্ত্রণ করে ভুল থেকে শেখে এবং সময়ের সাথে সাথে ট্রাফিক জ্যাম কমানোর চেষ্টা করে।

## ৫. Reward (পুরস্কার) কীভাবে ক্যালকুলেট করা হয়?

পুরস্কার বা Reward হলো একটি নম্বর যা AI-কে জানায় তার সিদ্ধান্ত কতটুকু ভালো ছিল। কোডে পুরস্কারটি ৪টি বিষয়ের ওপর ভিত্তি করে হিসেব করা হয়:

- **অপেক্ষার সময় কমা (Waiting Time reduction):** আগের মুহূর্তের চেয়ে বর্তমান মুহূর্তে রাস্তার মোট অপেক্ষার সময় (Waiting Time) কমলে AI পুরস্কার পায়।
- **জ্যাম কমা (Queue Length reduction):** লাইনে দাঁড়িয়ে থাকা গাড়ির সংখ্যা কমলে AI অতিরিক্ত পুরস্কার পায়।
- **অপেক্ষার জন্য জরিমানা (Waiting Penalty):** যদি অনেক বেশি গাড়ি অনেকক্ষণ ধরে অপেক্ষা করে, তবে AI থেকে পয়েন্ট কাটা হয়।
- **জ্যামের জন্য জরিমানা (Queue Penalty):** যদি রাস্তার লাইন খুব লম্বা হয়ে যায়, তবে তার জন্যও পয়েন্ট কাটা হয়।

**মূল লক্ষ্য:** জ্যাম কমানো এবং গাড়িগুলোর অপেক্ষার সময় সর্বনিম্ন পর্যায়ে রাখা।

## ৬. নিউরাল নেটওয়ার্ককে কী কী ডেটা (State) দেয়া হয়?

AI যেন সঠিক সিদ্ধান্ত নিতে পারে, সেজন্য তাকে প্রতিবার রাস্তার বর্তমান অবস্থা বা 'State' জানানো হয়। প্রতিটি লেনের (Lane) জন্য ৪টি প্রধান তথ্য দেয়া হয়:

১. **Halting Number:** ওই লেনে কতগুলো গাড়ি একদম স্থির হয়ে দাঁড়িয়ে আছে (জ্যাম বা লাল বাতির কারণে)।
২. **Waiting Time:** ওই লেনের গাড়িগুলো মোট কত সময় ধরে অপেক্ষা করছে।
৩. **Vehicle Number:** ওই লেনে বর্তমানে কতগুলো গাড়ি আছে (চলন্ত এবং স্থির সব মিলিয়ে)।
৪. **Mean Speed:** ওই লেনের গাড়িগুলোর গড় গতিবেগ কত।

এই তথ্যগুলো থেকে নিউরাল নেটওয়ার্ক বুঝতে পারে কোন রাস্তায় জ্যাম বেশি এবং কোন সিগন্যালটি এখন সবুজ করা উচিত।

## ৭. ইয়েলো ও গ্রিন টাইমের মান কেন সেট করা হয়?

কোডে `yellow_time = 3` এবং `green_time = 10` সেট করা হয়েছে। এর কারণগুলো হলো:

- **ইয়েলো টাইম (Yellow Time):** বাস্তবে যেমন ট্রাফিক লাইট সরাসরি সবুজ থেকে লাল হয় না, তেমনি সিমুলেশনেও একটি দিক বন্ধ করে অন্য দিক খোলার আগে কয়েক সেকেন্ড ইয়েলো লাইট জ্বালাতে হয়। এটি 'নিরাপত্তা' (Safety) নিশ্চিত করে এবং গাড়িগুলোকে থামার বা রাস্তা খালি করার সময় দেয়। এর ফলে সিমুলেশনে গাড়িগুলোর মধ্যে ধাক্কা লাগার সম্ভাবনা কমে।
- **গ্রিন টাইম (Green Time):** AI যখন কোনো একটি লেনকে সবুজ করার সিদ্ধান্ত নেয়, তখন সেই সিদ্ধান্তটি অন্তত কিছু সময়ের জন্য (যেমন: ১০ সেকেন্ড) স্থায়ী হওয়া উচিত। যদি সিদ্ধান্তটি বারবার (প্রতি সেকেন্ডে) বদলানো হয়, তবে ট্রাফিক জ্যাম আরও বাড়বে কারণ কোনো গাড়িই ঠিকমতো চৌরাস্তা পার হতে পারবে না। ১০ সেকেন্ড সময় দিলে গাড়িগুলো গতিপ্রাপ্ত হয় এবং চৌরাস্তাটি কার্যকরভাবে পার হতে পারে। এই সময়ের পর AI আবার নতুন করে অবস্থা (State) পর্যবেক্ষণ করে পরবর্তী সিদ্ধান্ত নেয়।